



Adaptive Pre-Distortion for Power Amplifier Linearization based on Neural Networks

Alexis Falempin

► To cite this version:

Alexis Falempin. Adaptive Pre-Distortion for Power Amplifier Linearization based on Neural Networks. Signal and Image processing. Université Grenoble Alpes [2020-..], 2022. English. NNT : 2022GRALT103 . tel-04032588

HAL Id: tel-04032588

<https://theses.hal.science/tel-04032588>

Submitted on 16 Mar 2023

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

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

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ GRENOBLE ALPES

École doctorale : EEATS - Electronique, Electrotechnique, Automatique, Traitement du signal (EEATS)

Spécialité : SIGNAL IMAGE PAROLE TELECOMS

Unité de recherche : CEA/LETI

Pré-Distorsion Adaptative basée sur des réseaux de neurones pour la Linéarisation d'amplificateurs de puissance

Adaptive Pre-Distortion for Power Amplifier Linearization based on Neural Networks

Présentée par

Alexis FALEMPIN

Encadrement de thèse :

Dr. Emilio CALVANESE STRINATI

Ingénieur de recherche (HDR), Université Grenoble Alpes,

Dr. Jean-Baptiste DORÉ

Ingénieur de recherche, Université Grenoble Alpes,

Dr. Rafik ZAYANI

Ingénieur de recherche (HDR), Université Grenoble Alpes

Directeur de Thèse

Co-encadrant de thèse

Co-directeur de thèse

Rapporteurs :

Pr. Yves LOUËT

Professeur, Centrale Supélec,

Dr. Myriam ARIAUDO

Maître de conférences (HDR), ENSEA

Thèse soutenue publiquement le **14 décembre 2022**, devant le jury composé de :

Pr. Yves LOUËT

Professeur, Centrale Supélec,

Dr. Myriam ARIAUDO

Maître de conférences (HDR), ENSEA

Pr. Laurent ROS

Professeur des Universités, Grenoble-INP

Dr. Marwa CHAFII

Professeure Associée, New York University Abu Dhabi

Rapporteur

Rapporteuse

Président

Examinatrice

Invités :

Dr. Lina MROUEH

Professeure, ISEP Paris,

Dr. Sylvain TRAVERSO

Ingénieur de recherche, Thales



À mes parents
Gilles Falempin et Rosa Maria Brunet Viñes,

À la mémoire de ma mère
Rosa Maria Brunet Viñes,
*Puisses-tu reposer en paix,
et être fière de tes enfants.*

Abstract

MODERN wireless communications are pushing the limits of data rate and communication bandwidth needs. With the appearance of new services, users are expected to increase their data usage and access to services, requiring instantaneous response time. To that extent, 5G and future wireless technologies are in the race for high data rate communications, low-latency and connectivity of billions of things, offering new services for users. However, these improvements lead to key challenges, namely sustainability, energy efficiency and overall system complexity. Looking at the future, it appears that for 6G technology, these challenges will become primordial. Specifically, particular attention shall be paid to offer a sustainable design and development of transmitters and receivers hardware components, which are exhibiting key issues about energy consumption. Thus, the design of transceivers must be rethought to put forward low-complexity and sustainable solutions.

In this thesis, our objective is to enhance the power efficiency of radio-frequency power amplifiers, leading to a high global system energy efficiency. Indeed, the power amplifier represents the most power hungry hardware component of the transmitter chain. Then, its power efficiency must be maximized to improve global carbon footprint. Nonetheless, this hardware component exhibits strong nonlinearities when operated close to its saturation, where its power efficiency is high. Therefore, we focus on digital pre-distortion approach to linearize the power amplifier and improve its power efficiency. To propose an efficient linearizing module, deep learning, neural networks, solutions have emerged as they offer robustness and reliability to solve nonlinear problems. However, neural networks and particularly applied to digital pre-distortion are facing several issues which represent a bottleneck. Complexity, adaptivity and energy efficiency are the main aspects that a pre-distortion module must take into consideration to be robust. These represent the main challenges of neural networks in this use case. To address these challenges, we propose in this thesis, a novel design of digital pre-distortion based on neural networks to linearize efficiently a power amplifier.

To achieve such goal, this thesis is structured on three main axes. (1) First, we propose a fully customized architecture of neural networks dedicated to perform digital pre-distortion. By doing so, we think differently from the “blackbox” architecture design and drastically lower the complexity of the digital pre-distortion module. By breaking down the complexity, we are able to offer a cost-effective digital pre-distortion. (2) Second, we investigate approaches to deal with the time-varying effects that may affect the power amplifier, such as the temperature or electrical variations. Usually, neural networks have a poor capacity to adapt to unseen events during their training. Then, to improve the adaptivity of our solution, we choose to use a new class of learning algorithms called meta-learning. Using the latter, we demonstrate that with few data and calibration time, our proposed digital pre-distortion is able to deal with time-varying power amplifiers, with satisfying results for real-world applications. (3) Third, we further optimize our module by considering hardware implementation aspects. We propose to perform quantization on our neural networks to improve the computing resource usage and lower the energy consumption. This step is crucial to propose a hardware implementation since we would like to optimize the computing load and energy efficiency of the system. To avoid the noise induced by the quantization operation, we learn our model how to mitigate this noise by using a quantization aware training. Our results show that our quantized digital pre-distortion has the same performance as without quantization, and presents the same complexity as the implementation of a fast Fourier transform.

Keywords – 5G and beyond Networks, Wireless Networks, Power Amplifiers, Digital Pre-Distortion, Energy Efficiency, Machine Learning, Neural Networks, Meta-Learning, Quantization.

Résumé

LES technologies de communications sans-fil modernes repoussent les limites actuelles en termes de débit et bande passante. En effet, l'apparition de nouveaux services impose un débit et accès fortement accrus. La 5G et les technologies de communications futures font l'objet d'une course au haut-débit, à la faible latence et à la connexion de milliards d'objets. Cependant, de telles améliorations impliquent pléthore de défis à relever tels que la durabilité, l'efficacité énergétique, et la complexité des systèmes. A long terme, il semble que pour la 6G, ces défis deviendront primordiaux. En l'occurrence, une attention particulière devra être portée à la conception durable des éléments matériels constituant les transmetteurs et récepteurs sans-fil. Ainsi, il convient de repenser leur conception pour mettre avant des solutions basse complexité et durables. Cette thèse a pour objectif d'améliorer l'efficacité énergétique des amplificateurs de puissance radiofréquence. L'amplificateur de puissance est le composant le plus énergivore de la chaîne de transmission radiofréquence. Ainsi, son efficacité énergétique doit être maximisée pour améliorer l'empreinte carbone du système. Mais, ce composant matériel présente des non-linéarités quand il est utilisé proche de sa saturation, zone où son efficacité est maximale. Par conséquent, dans cette thèse, nous nous intéressons à la pré-distorsion numérique pour linéariser l'amplificateur de puissance, et ainsi améliorer son efficacité énergétique. Afin de proposer un module de linéarisation efficace, des techniques d'apprentissage profond, reposant sur des réseaux de neurones, sont considérées. En effet, elles offrent une certaine robustesse et fiabilité pour résoudre des problèmes non linéaires. Toutefois, les réseaux de neurones et particulièrement appliqués à la pré-distorsion numérique, présentent plusieurs problèmes. La complexité, l'adaptabilité et l'efficacité énergétique sont les principaux aspects à prendre en compte pour concevoir une pré-distorsion efficace ; ces derniers faisant défaut aux réseaux de neurones. Pour traiter ces défis, nous proposons une nouvelle conception de pré-distorsion numérique basée sur des réseaux de neurones pour linéariser efficacement l'amplificateur de puissance.

Pour atteindre un tel but, cette thèse se construit autour de trois grands axes. (1) Tout d'abord, nous proposons une architecture de réseaux de neurones spécifique et dédiée à la conception d'une fonction de pré-distorsion. Ce faisant, nous améliorons drastiquement la complexité de cette dernière. (2) Ensuite, nous abordons des approches permettant de faire face aux contraintes temporelles qui influent sur l'amplificateur de puissance. On peut notamment citer des effets relatifs à la température et variations électriques du composant. Généralement, les réseaux de neurones sont peu performants lorsqu'ils doivent s'adapter à des événements inconnus de leur apprentissage. Ainsi, pour améliorer l'adaptabilité de notre solution, nous employons une classe d'algorithmes nommée "méta-apprentissage". Par ce biais, nous démontrons que notre système peut s'adapter en utilisant peu de données et temps de calibration avec des résultats satisfaisants, permettant ainsi son usage en condition réelle. (3) Finalement, nous optimisons notre module en vue d'une implémentation matérielle. Spécifiquement, nous proposons d'agir sur la quantification de nos réseaux de neurones pour améliorer leurs consommations en ressources et énergie. Cette opération s'avère cruciale pour une implémentation où les ressources s'avèrent limitées. Pour éviter le bruit induit par la quantification, nous apprenons à notre solution comment l'atténuer. Nos résultats montrent que l'emploi de la quantification n'altère pas les performances du système. De plus, la solution quantifiée présente la même complexité matérielle qu'une transformée de Fourier.

Mots-clés – Réseaux sans fil, Amplificateur de Puissance, Pré-Distorsion Numérique, Efficacité Énergétique, Apprentissage Automatique, Réseaux de Neurones, Méta-apprentissage, Quantification.

List of Author's Publications

The content of this manuscript is based on the following patents, conferences and journal publications.

Patents

- [P1] **A. Falempin**, R. Zayani and J-B. Doré, “Device for linearizing a power amplifier of a communication system by digital predistortion,” Issued in July 05, 2022, US2107230.

International Journal Communications

- [J1] **A. Falempin**, J-B. Doré, R. Zayani and E. Calvanese Strinati, “Low-Complexity Adaptive DPD: From online optimization to meta-learning,” IEEE Transactions on Broadcasting, pp:pp, 2022.
- [J2] **A. Falempin**, J.Laurent, J-B. Doré, R. Zayani and E. Calvanese Strinati, “Implementation of Neural Networks based Digital Predistortion for PA linearization in OFDM systems,” **To be submitted before 2023**, 2023.

International Conference Communications

- [C1] E. Calvanese Strinati, D. Belot, **A. Falempin** and J-B. Doré, “Toward 6G: From New Hardware Design to Wireless Semantic and Goal-Oriented Communication Paradigms,” In Proc. IEEE European Solid State Circuits Conference (ESSCIRC), Grenoble, FR, pages 275-282., 2021.
- [C2] **A. Falempin**, R. Zayani, J-B. Doré and E. Calvanese Strinati, “Low-Complexity Adaptive Digital Pre-Distortion with Meta-Learning based Neural Networks,” In Proc. IEEE Consumer Communications & Networking Conference (CCNC), NV, USA, pages 453–457., Feb 2022.
- [C3] **A. Falempin**, J. Laurent, J-B. Doré, R. Zayani and E. Calvanese Strinati, “On the Performance of Quantized Neural Networks based Digital Predistortion for PA linearization in OFDM systems,” In Proc. IEEE Conference on Vehicular Technology (VTC2022-Fall), London, UK, pages 453–457., 2022.

Extra-thesis collaborations

These are the results of some collaborations done during the thesis but not directly related to it.

International Conference Communications

- [C4] **A. Falempin**, J. Schmitt, T.D. Nguyen and J-B. Doré, “Towards Implementation of Neural Networks for Non-Coherent Detection MIMO systems,” In Proc. IEEE Conference on Vehicular Technology (VTC2022-Fall), London, UK, 2022.

International Journal Communications

- [J3] S. Bicaïs, **A. Falempin**, J-B. Doré and V. Savin, “Design and Analysis of MIMO Systems Using Energy Detectors for Sub-THz Applications,” IEEE Transactions on Wireless Communications, 3678-3690, 2022.

Contents

List of Figures	ix
List of Tables	x
List of Acronyms	xi
List of Symbols	xiv
I Generalities	1
1 Introduction	2
1.1 The race to 5G and beyond networks	3
1.1.1 The toolbox to win the race	3
1.1.2 The century challenge: Sustainable networks	4
1.2 Power amplifiers: What's the catch?	4
1.2.1 Nonlinearities	5
1.2.2 Power efficiency	5
1.2.3 Optimizing the power amplifier efficiency	5
1.3 Machine learning for communications	6
1.4 Thesis outline and Contributions	8
2 Digital Pre-Distortion for Power Amplifiers	10
2.1 Introduction	11
2.2 Communication system model	11
2.3 The Power Amplifier: Essential but problematic	12
2.3.1 An insight on nonlinearities	12
2.3.2 Power amplifier modeling	12
2.3.3 Performance metrics	16
2.3.4 Power efficiency	17
2.4 Digital Pre-Distortion: Linearizing a Power Amplifier	19
2.4.1 DPD modeling	19
2.4.2 Indirect Learning Architecture (ILA)	20
2.4.3 State-of-the-art approaches	20
2.5 Conclusion	21
II Efficient AI-based Digital Pre-distortion solutions for PA Linearization	22
3 Enabling Low-Complexity Neural Networks for Digital Pre-Distortion	23
3.1 Introduction	23
3.1.1 Motivations	24
3.1.2 Related Work	24
3.1.3 Contributions	24
3.2 Background on Deep Learning	25
3.2.1 Theory	25
3.2.2 Toy example: the sine function	28

3.3	Deep Learning for Digital Pre-Distortion	30
3.3.1	State-of-the-art approach: Cartesian DPD	30
3.3.2	Towards a Low-Complexity DPD Neural Network (LCDPDNN)	30
3.3.3	Numerical simulations	33
3.4	Conclusion	40
4	Learning how to Learn the Digital Pre-Distortion	41
4.1	Introduction	42
4.1.1	Motivations	42
4.1.2	Related Work	42
4.1.3	Contributions	43
4.2	Meta-Learning : A new way of optimizing neural networks	43
4.2.1	Model-Agnostic Meta-Learning (MAML) for DPD	43
4.2.2	Dataset construction for DPD	47
4.2.3	Numerical simulations	48
4.2.4	Complexity	50
4.3	Efficient Weights Selection for Adaptive DPD	51
4.3.1	DPD Neural Network Weights Selector (DPDNNWS)	51
4.3.2	Numerical simulations	52
4.3.3	Complexity	55
4.4	Comparison between DPD algorithms	55
4.4.1	Spectral analysis and ACLR	55
4.4.2	Synthesis on all performance metrics	56
4.5	Conclusion	56
5	Quantized Neural Network based DPD for Efficient Hardware Implementation	58
5.1	Introduction	59
5.1.1	Motivations	59
5.1.2	Related Work	59
5.1.3	Contributions	60
5.2	Quantized Neural Network based DPD	60
5.2.1	Number representation	60
5.2.2	Post-Training Quantization (PTQ)	62
5.2.3	Quantization-Aware Training (QAT)	63
5.2.4	Numerical simulations	64
5.3	Towards FPGA implementation	68
5.3.1	FPGA architecture and programming	68
5.3.2	Numerical simulations	69
5.4	Conclusion	71
6	Conclusion and Perspectives	72
6.1	Overall conclusion	73
6.2	Short-term perspectives	74
6.2.1	Towards a realistic implementation	74
6.2.2	Integrating memory effects of the PA	74
6.3	Mid-term perspectives	74
6.3.1	Joint PAPR reduction and DPD	74
6.3.2	Complex-valued Neural Networks for communications	75
6.4	Long-term perspectives	75
6.4.1	Massive MIMO DPD	75

6.4.2	Energy-efficient communications	76
6.4.3	Meta-learning for communications	76
Appendices		77
A	Résumé étendu de thèse	78
A.1	Chapitre 1 : Introduction	78
A.1.1	La boîte à outils du champion	78
A.1.2	Le défi du siècle : Des réseaux durables	79
A.1.3	Les amplificateurs de puissance	80
A.1.4	Apprentissage automatique pour les télécommunications	81
A.2	Chapitre 2 : La pré-distorsion numérique pour amplificateurs de puissance	81
A.2.1	Modélisation du PA	81
A.2.2	Pré-distorsion numérique : Linéariser un PA	82
A.3	Chapitre 3 : DPD basse complexité à base de réseaux de neurones (LCDPDNN)	83
A.3.1	Architecture des réseaux de neurones pour la DPD	83
A.3.2	Analyse de performances	84
A.3.3	Contributions	84
A.4	Chapitre 4 : Apprendre à apprendre la DPD	85
A.4.1	Méta-apprentissage: MAML	85
A.4.2	Sélection de poids (DPDNNWS)	85
A.4.3	Analyse de performances	86
A.4.4	Contributions	87
A.5	Chapitre 5 : Quantification de la DPD pour une implémentation matérielle efficace	87
A.5.1	Représentation d'un nombre	87
A.5.2	Quantification post-entraînement de la DPD (PTQ)	87
A.5.3	Apprendre à corriger l'erreur de quantification (QAT)	88
A.5.4	Analyse des performances	88
A.5.5	Implémentation sur FPGA	88
A.5.6	Contributions	89
A.6	Conclusion	89
B	Design and Analysis of MIMO Systems Using Energy Detectors for Sub-THz Applications	90
C	Towards Implementation of Neural Networks for Non-Coherent Detection MIMO systems	104
Bibliography		110

List of Figures

1.1	Energy consumption of an operator	4
1.2	Power consumption of the radio processing	5
2.1	OFDM system with a DPD, a PA and a loopback link	11
2.2	Received 16-QAM constellation using a PA	12
2.3	General characteristics of the power amplifier	13
2.4	AM/AM and AM/PM characteristics of custom Rapp PA model	15
2.5	Variation of PA gain	16
2.6	Power Spectral Density (PSD) of OFDM	17
2.7	Impact of a high peak amplitude on the PA PAE	18
2.8	Transfer function of DPD, PA and resulting system	20
2.9	Indirect Learning Architecture	20
3.1	Generic NN architecture	25
3.2	Learning process of a NN	27
3.3	Fitting of sine function	29
3.4	Classic NN based DPD	30
3.5	Architecture of the LCDPDNN for AM/AM and AM/PM distortions	31
3.6	System model with DPD in real-time	33
3.7	AM/AM and AM/PM characteristics using with PA and DPD	35
3.8	EVM performance in (dB) of LCDPDNN and literature solutions	36
3.9	PSD before and after LCDPDNN (CL) using an oversampled FFT, $p = 0.7$	37
3.10	PSD before and after LCDPDNN (CL) with a digital up converter, $p = 0.7$	38
3.11	PSD using presented DPD and state-of-the-art solutions with $IBO_{dB} = 8\text{dB}$, $p = 0.7$	38
3.12	EVM performance in (dB) with $0.7 \leq p \leq 1.5$, $IBO = 8\text{dB}$	39
4.1	Representation of PA tasks with $p \in [0.7, 1.5]$	45
4.2	MAML algorithm state machine	46
4.3	EVM performance in (dB) with $0.7 \leq p \leq 1.5$, $IBO_{dB} = 8\text{dB}$	49
4.4	PSD before and after LCDPDNN (MAML) $p = 0.7$, $IBO_{dB} = 8\text{dB}$	50
4.5	EVM performance in (dB) of DPDNNWS and MAML, $IBO_{dB} = 8\text{dB}$	53
4.6	EVM performance in (dB) using DPD chosen by the DPDNNWS, $IBO = 8\text{dB}$	54
4.7	PSD using DPD (CL) and DPDNNWS with $p = 0.7$, $IBO_{dB} = 8\text{dB}$	54
4.8	PSD comparison using presented algorithms and literature solutions with $IBO = 8\text{dB}$, $p = 0.7$	55
5.1	QAT algorithm for one gradient step	64
5.2	EVM performance in (dB) using PTQ and QAT	66
5.3	PSD using PTQ and QAT, $N_{bits} = 10$	67
5.4	PSD using PTQ and QAT, $N_{bits} = 4$	67
5.5	FGPA Architecture of NN performing AM/AM DPD	69
5.6	Architecture of NN performing AM/PM DPD	70
5.7	Field-Programmable Gate Array (FPGA) resource consumption	71
A.1	Consommation énergétique des opérateurs	79
A.2	Puissance consommée par le traitement radio	80
A.3	Impact d'une excursion de signal élevée sur la PAE du PA	83

A.4	Architecture du LCDPDNN pour les distorsions AM/AM et AM/PM	84
A.5	Ensemble de tâches de PA $p \in [0.7, 1.5]$	86
A.6	Performance en utilisant PTQ et QAT	88

List of Tables

2.1	EVM requirements for LTE	17
3.1	Common activation functions	26
3.2	Hyperparameters for sine fitting	28
3.3	Parameters for physical layer	34
3.4	Hyperparameters for DPD learning	35
3.5	ACLR in (dB)	37
3.6	Comparison of DPD algorithms	39
4.1	Meta-learning parameters	48
4.2	ACLR in (dB)	56
4.3	Comparison with state-of-the-art	57
5.1	Comparison between floating-point and fixed-point	62
5.2	Quantization of NN weights and activations	65
A.1	Comparaison d’algorithmes de DPD	84
A.2	Comparaison avec l’état de l’art	86

List of Acronyms

1G	First Generation
2G	Second Generation
3G	Third Generation
4G	Fourth Generation
5G	Fifth Generation
6G	Sixth Generation
ACLR	Adjacent Channel Leakage Ratio
ADC	Analog-to-Digital Converter
AI	Artificial Intelligence
AM/AM	Amplitude-to-Amplitude
AM/PM	Amplitude-to-Phase
BRAM	Block RAM
C2P	Cartesian-to-Polar
CLB	Configurable Logic Block
CNN	Convolutional Neural Network
CVNN	Complex-Valued Neural Network
DAC	Digital-to-Analog Converter
DDC	Digital Down Conversion
DLA	Direct Learning Architecture
DPD	Digital Pre-Distortion
DPDNNWS	DPD Neural Network Weights Selector
DSP	Digital Signal Processing
DUC	Digital Up Conversion
eMBB	enhanced Mobile Broad-Band
EVM	Error Vector Magnitude
FCN	Fully Connected Network
FFT	Fast Fourier Transform
FIFO	First In First Out

FLOP	Floating-Point Operation
FPGA	Field-Programmable Gate Array
GD	Gradient Descent
GLRT	Generalized Likelihood Ratio Test
GPU	Graphics Processing Unit
GSM	Global System for Mobile Communications
IBO	Input Back-Off
IFFT	Inverse Fast Fourier Transform
ILA	Indirect Learning Architecture
IoT	Internet of Things
IP	Internet Protocol
IQ	In-Phase and Quadrature
LCDPDNN	Low-Complexity DPD Neural Network
LTE	Long Term Evolution
LSTM	Long-Short-Term-Memory
LUT	Look-up Table
MAC	Medium Access Control
MAML	Model-Agnostic Meta Learning
MLE	Maximum Likelihood Estimation
mmWave	millimeter-Wave
mMTC	massive Machine-Type Communications
MIMO	Multiple-Input Multiple-Output
MSE	Mean Squared Error
MTL	Mutli-Task Learning
NN	Neural Network
OFDM	Orthogonal Frequency Division Multiplexing
OOB	Out-of-Band
P2C	Polar-to-Cartesian
PA	Power Amplifier
PAE	Power Added Efficiency
PAPR	Peak-to-Average Power Ratio

PTQ	Post-Training Quantization
PSD	Power Spectral Density
QAM	Quadrature Amplitude Modulation
QAT	Quantization-Aware Training
RAN	Radio Access Network
ReLU	Rectified Linear Unit
RF	Radio-Frequency
SGD	Stochastic Gradient Descent
SISO	Single-Input Single-Output
SMS	Short Messaging Service
STE	Straight-Through Estimator
URLLC	Ultra-Reliable Low-Latency Communications
VHDL	Very High Speed Integrated Circuit Hardware Description Language
VR	Virtual Reality

List of Symbols

System parameters

h	Channel perturbation coefficient
N_{OFDM}	Number of OFDM symbols
N_{fft}	Size of FFT for OFDM modulation
N_{CP}	Size of cyclic prefix for OFDM
N_a	Number of active subcarriers
f_{sc}	Subcarrier spacing frequency
M	Modulation Order
B_W	Bandwidth
ρ	Related to power amplifier AM/AM distortion
Φ	Related to power amplifier AM/PM distortion
f_ρ	AM/AM characteristic of the power amplifier
f_Φ	AM/PM characteristic/Phase shift of the power amplifier
G	Small gain of the power amplifier
p	“knee factor” of Rapp model, impacts on model linearity

Communication signals

\mathbf{i}	OFDM symbols
\mathbf{x}	Input signal of power amplifier
\mathbf{y}	Output signal of power amplifier
\mathbf{z}	Received symbols
σ_w^2	Thermal noise variance

Mathematical Notations

x	Lowercase symbol represents a scalar number
\mathbf{x}	Bold lowercase symbol represents a vector
\mathbf{X}	Bold uppercase symbol represents a matrix
$\mathcal{N}(\mu, \sigma^2)$	Normal distribution with mean μ and variance σ^2
$\mathbb{CN}(\mu, \sigma^2)$	Complex normal distribution with mean μ and variance σ^2
$\mathbb{E}[\cdot]$	Expectation operator
$ \cdot $	Absolute value
$\arg(\cdot)$	Argument of complex number
f^{-1}	Inverse of f
θ	NN weights
\mathcal{D}	Dataset for neural network training
$\mathcal{L}(\mathcal{D}, \theta)$	Loss function between dataset \mathcal{D} and predictions of NN using θ
∇	Gradient operator
η	Learning rate

Part I

Generalities

Introduction

Contents

1.1	The race to 5G and beyond networks	3
1.1.1	The toolbox to win the race	3
1.1.2	The century challenge: Sustainable networks	4
1.2	Power amplifiers: What's the catch?	4
1.2.1	Nonlinearities	5
1.2.2	Power efficiency	5
1.2.3	Optimizing the power amplifier efficiency	5
1.3	Machine learning for communications	6
1.4	Thesis outline and Contributions	8

1.1 The race to 5G and beyond networks

THE world is witnessing a tremendous digitalization of every service, and particularly related to use of mobile devices. Since the past decades, wireless technologies are in the race to satisfy new needs of the growing society we live in. The starting point of this race began in the 80s with the First Generation (1G) of wireless communications, enabling voice calls. Back in time, this was an incredible breakthrough. Since then, a new generation of wireless communications is introduced nearly every ten years, pushing forward new services. In the 90s, the Second Generation (2G), namely Global System for Mobile Communications (GSM) [1], notably enabled the Short Messaging Service (SMS) and first data services alongside digital communications. Later on in 2001, Third Generation (3G) [2] systems are deployed enabling data rates around several megabits per second. 2009 marks the arrival of Long Term Evolution (LTE) systems [3], with a huge improvement in terms of data rates and latency, installing the Fourth Generation (4G) of wireless communications [4]. 4G communications notably enabled the access to instant messaging services, video streaming services and voice over Internet Protocol (IP). Notwithstanding, emerging new services, namely autonomous driving, telemedicine, Virtual Reality (VR) and Internet of Things (IoT) are in demand of high-data rates and low-latency. This incites a constant evolution of wireless networks to address all these new challenges. Recently developed and currently under deployment, Fifth Generation (5G) networks introduces a breakthrough in the design of communication networks. The new radio and core network architecture contribute to higher data rates, lower latency and connection of billions of devices, thus welcoming new verticals. Then, 5G is built around three scenarios [5]:

- **enhanced Mobile Broad-Band (eMBB)**: it addresses applications requiring high data rates, up to 20Gbps in downlink [6]. This scenario puts forward a wide-area coverage to provide continuous access and high capacity in dense areas;
- **Ultra-Reliable Low-Latency Communications (URLLC)**: this use case is totally disruptive compared to previous generations. It concerns new applications in need of optimized latency and availability. This may concern services such as remote medical surgery, smart grids and Industry 4.0;
- **massive Machine-Type Communications (mMTC)**: this specific scenario shall support the connectivity of billions of devices with small data usage and low-power consumption.

1.1.1 The toolbox to win the race

To address the aforementioned challenges, new technologies have to be considered both in physical layer and Medium Access Control (MAC) control. Here are some key enabling technologies. To achieve high-capacity of the wireless networks, *i.e.* tens of gigabits per second, 5G leverages millimeter-Waves (mmWaves) frequency spectrum which, for now, is mainly unused. mmWaves implies using higher frequencies (> 24GHz). However, high-frequency signal transmissions suffer from severe path loss, attenuation, blockage and directivity issues due to the shortened wavelength. Meanwhile, the emergence of massive Multiple-Input Multiple-Output (MIMO) technology permits alleviating the problems of mmWave thanks to large gain and high-directive antennas [7]. In conjunction with these technologies, the choice of the radio waveform has a significant impact on the Radio Access Network (RAN) since it particularly affects the transceiver, *i.e.* transmitter and receiver, design and complexity. For instance, since LTE, Orthogonal Frequency Division Multiplexing (OFDM) is the retained waveform for the RAN [8]. OFDM, first proposed in [9], brings up many advantages such as resistance to frequency selective fading and inter symbol interference. On top of that, its implementation is computationally efficient by using Inverse Fast Fourier Transform (IFFT) and Fast Fourier Transform (FFT). However, this waveform present a high Peak-to-Average Power Ratio (PAPR) which challenges the transceivers

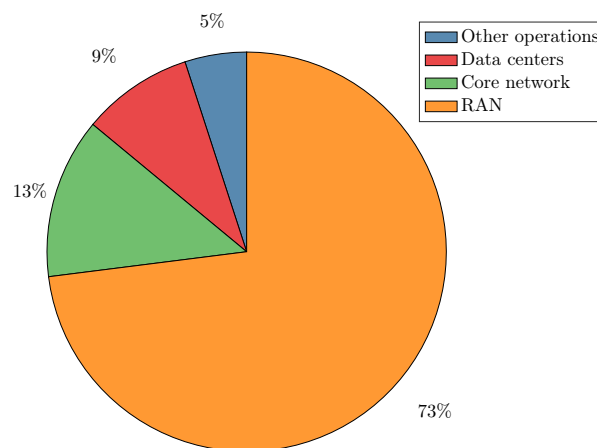


Figure 1.1: Energy consumption of an operator

designs, and specially the Power Amplifier (PA). These technologies contribute in part to the infinite race of performance in terms of bandwidth, latency and so on. However, all these improvements enforce complex design of transceivers which are the key components of transmission and reception chains in a wireless network. Particularly, important challenges are to be considered at the transceivers, and notably the energy efficiency.

1.1.2 The century challenge: Sustainable networks

While the race towards performance is moving forward, a major challenge is to be addressed, and it concerns the energy efficiency of wireless technologies. Since the beginning of wireless technologies, power consumption has been neglected in favor of the performance. 5G wireless network starts considering the problem of energy-efficiency thanks to new technologies [10] including massive MIMO, sleep modes and machine learning. Eventually, future communications technologies, such as Sixth Generation (6G) technology, have a roadmap centered around the energy-efficiency and the sustainability [11]. The golden question is then: *How to improve the overall energy-efficiency of wireless networks?* The answer to this question requires analyzing the power consumption of the end-to-end wireless communications systems.

Figure 1.1 presents the average energy consumption of operators. This measurement has been conducted in [12] and concerns 31 different networks. We can see that most of the energy consumption comes from the RAN. The RAN consumption includes the energy consumed by the base station, NodeB, eNodeB and gNodeB. Then, it appears that the most power hungry element in the network is the base station. More specifically, Figure 1.2a shows the power consumption of a base station, and Figure 1.2b analyzes the power consumption of an antenna unit in the base station. These measurements are extracted by the NGMN Alliance report on energy efficiency [13].

From all these pie charts, we can state that the main optimizations in terms of energy efficiency shall be done at the base station. Specifically, we can see that the PA consumes 59% of the power allocated to an antenna unit. Hence, it becomes important to improve its power efficiency since it is the most power hungry hardware component in the network.

1.2 Power amplifiers: What's the catch?

A PA is a hardware component which converts a low magnitude signal into a high magnitude signal. Ideally, the PA should apply a gain to its input resulting in a linear amplified output. However, PAs are nonlinear because they contain transistors that present nonlinear behaviors. Admittedly, the PA will

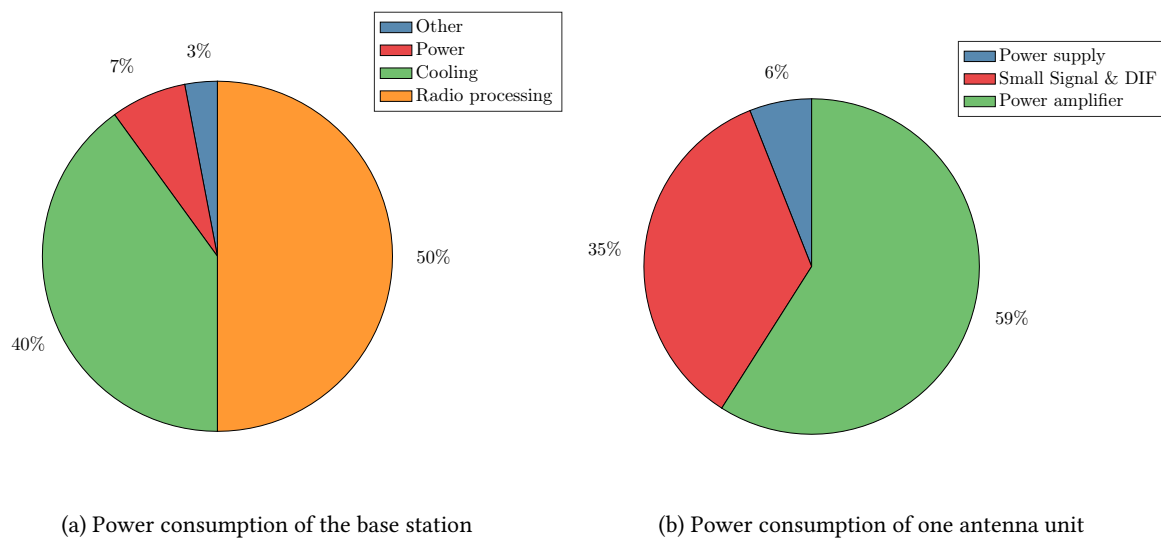


Figure 1.2: Power consumption of the radio processing

amplify the incoming signal, but distortions will also be injected resulting in detection errors at the receiver side.

To understand why research is needed around the PA efficiency, we present here some insights on the impact of the use of both OFDM and PA in the Radio-Frequency (RF) chain.

1.2.1 Nonlinearities

Operating the PA with high power efficiency, *i.e.* in its nonlinear region, leads to severe distortions affecting the transmission quality. Therefore, these distortions should be tackled to enhance the system performance. Then, we must tune a trade-off between nonlinearities and power efficiency of the PA.

1.2.2 Power efficiency

The power efficiency of the PA is maximum near its saturation. However, near the saturation, the PA also exhibits severe nonlinearities which will degrade the received signal as in Figure 2.2. As explained in the previous paragraph, the high PAPR of OFDM will lead to a saturation of the PA. To avoid this problem, it is often considered scaling down the average power of the signal [14]. This operation, called back-off, allows to reach the linear regime of the PA, avoiding then the nonlinearities. However, by doing so, the power efficiency of the hardware component drops at its minimum.

Then, to enhance global system energy efficiency, a joint optimization of the power consumption and nonlinearities must be conducted. Namely, one shall correct the nonlinearities of the PA, *i.e.* linearizing the PA, while keeping its maximum efficiency.

1.2.3 Optimizing the power amplifier efficiency

To leverage the maximum efficiency of the PA, research is currently focused on two main aspects: i) PAPR reduction of waveforms, and ii) PA linearization.

1.2.3.1 PAPR reduction

One way to improve the PA efficiency is to minimize the back-off operation. Doing so will improve the PA efficiency since it could be operated at higher input powers without saturating. To achieve such goal, a common technique is to reduce the PAPR of waveforms, leading to less power fluctuations of

the signal. For instance, considering OFDM systems, multiple studies have been conducted to address the problem of PAPR. Namely, clipping, partial transmit sequence, and selected mapping are proven to successfully reduce the PAPR of the OFDM [15]. Alternatively, it can also be considered to optimize the constellation [16] [17] benefiting from a reduced complexity compared to classical PAPR reduction schemes. Besides, studies are also conducted to propose new waveforms having a lower PAPR compared to OFDM [18] [19].

Thus, all these different techniques allow using waveforms with lower PAPR, increasing then the PA efficiency thanks to a reduced back-off. However, even if the PA can be operated near its saturation, it will still exhibit nonlinearities.

1.2.3.2 PA linearization

Otherwise, to cope with nonlinearities exhibited by the power amplifier, many techniques have been proposed in order to linearize the PA [20]. It exists mainly three linearization techniques: feedforward, feedback and pre-distortion. We give here some insights on each technique. A more detailed analysis is given in Chapter 2.

Feedforward: This technique has been proposed in 1927 [21], and represents the first linearization technique. Its concept is simple but complex to implement. The idea is to get the distortion error by subtracting the attenuated output of the PA, to linearize, with the input signal. Then, the distortion error is amplified by a linear amplifier with the gain as the PA exhibiting nonlinearities. Finally, the resulting distortion is subtracted to the output of the PA, resulting in a linear system. Delays are required to synchronize the linearization because one needs to wait the output of the amplifiers. This technique can be complex since the delays must be tightly tuned. Moreover, it causes some issues regarding global energy efficiency because two PAs are needed.

Feedback: This technique has also been proposed in [21] to improve the feedforward linearization. The idea is to use a loopback link to detect the error between the output of the PA and the input, such that this error can be minimized by subtracting it to the input. This method can be used either on the polar representation of the signals or their Cartesian representation [22]. The main drawback of this technique is the delay needed in the loopback link which makes difficult to linearize wideband signals.

Pre-distortion: Over the years, it has become the most common used technique to linearize a PA. One of the earliest implementation of this technique has been conduct in the 1970' [23]. Pre-distortion consists in adding a module before the PA such that the resulting system, *i.e.* pre-distortion plus PA, is linear. Then, the PA compensates the nonlinearities brought by the pre-distortion. Thus, the pre-distortion consists in finding the inverse of the distortions induced by the PA. This can be achieved using different techniques further described in this thesis. Nonetheless, since the appearance of this technique, research is still ongoing to solve challenges related to the evolution of the telecommunications. Namely, complexity and energy efficient are still challenging while designing pre-distortion functions. On top of that, the PA is subject to time-varying effects such as aging, temperature, electrical variations, *etc.* Then, ideally, pre-distortion shall be able to cope with these effects as well to maximize the global energy efficiency.

1.3 Machine learning for communications

In the previous paragraphs, we have seen that the evolution towards new communications technologies implies new challenges to be solved. The latter are often highly nonlinear and non-convex problems that may not have a straightforward solution using analytical tools. Thus, machine learning techniques have emerged to help solving this kind problem. In a few words, machine learning can be represented

as, a class of algorithms that learn automatically how to solve a specific problem by using data related to it. In particular, deep learning, a subfield of machine learning, has proven to be efficient at solving nonlinear problems. Thus, in wireless communications this kind of techniques can be used for physical layer issues such as channel estimation, detection, noise mitigation, *etc*[24] [25].

Besides, it has also been considered to optimize the PA efficiency with Neural Networks (NNs), notably by proposing PAPR reduction solutions [26] or pre-distortions solutions [27]. Yet, it is also known that deep learning techniques often exhibit high complexity, and are difficult to implement on real hardware. Then, it becomes important to deal with those aspects since sustainable and cost-effective communications are foreseen for future technologies. Thus, in this thesis, we will consider the use of NNs to perform Digital Pre-Distortion (DPD) for PA linearization taking into account the aforementioned aspects.

1.4 Thesis outline and Contributions

The objective of this thesis is to propose an energy efficient pre-distortion function for PA linearization. Thus, in chapter 2, we give some theoretical elements about PA modeling and DPD techniques. Particularly, in this chapter, we show that deep learning techniques can bring real benefits to perform DPD compared to classical techniques. Then, starting from chapter 3, we present our research work conducted to break down the hardware complexity of the DPD, while presenting good linearization performance in multiple scenarios. Each chapter presents a dedicated state-of-the-art for clarity's sake.

Chapter 3: Breaking down the complexity of NN based DPD. In chapter 3, we propose a novel design of DPD using NNs. Specifically, we tackle separately the distortions induced by the PA, namely amplitude and phase distortions. By doing so, we propose dedicated and custom NN architectures correcting amplitude and phase nonlinearities respectively. The designed solutions rely on an analysis of PA characteristics to propose adapted and customized architectures. Contrary to the “black-box”, *i.e.* wide fully connected architectures, paradigm often encountered in NN design, our designs are fully tailored to the DPD use case. Leveraging explainable NN models, as we propose here, allows to drastically reduce the complexity of the solution. For instance, considering the DPD, our proposed NNs achieve low-complexity while satisfying the performance requirements for LTE and 5G standards. Therefore, this chapter aims to provide the best trade-off between complexity and performance to linearize the PA efficiently. Eventually, the technical novelties presented here have been disseminated in the following contributions.

[C1] A. Falempin, J-B. Doré, R. Zayani and E. Calvanese Strinati, “Low-Complexity Adaptive Digital Pre-Distortion with Meta-Learning based Neural Networks,” In Proc. IEEE Consumer Communications & Networking Conference (CCNC), NV, USA, pages 453–457., Feb 2022.

[J1] A. Falempin, J-B. Doré, R. Zayani and E. Calvanese Strinati, “Low-Complexity Adaptive DPD: From online optimization to meta-learning,” IEEE Transactions on Broadcasting, 2022.

Chapter 4: Learning how to quickly deal with time-varying effects. Power amplifiers are subject to their environment, and multiple time-varying effects may affect their behavior. In particular, temperature and electrical variations has a direct impact on the PA characteristics. Thus, in chapter 4, we study new approaches to adapt the DPD, proposed in chapter 3, to a change of the PA state. First, we propose to use a meta-learning algorithm that allows learning the best initialization for the DPD, such that it can quickly adapt to a new state of the PA. This approach works by learning from multiple known characterizations of the PA to propose a generalized model. Using this approach enables an online adaptation of the DPD using few data and reduced calibration time. For online adaptation, we also propose an algorithm to select pertinent data to conduct the learning. By doing so, we greatly improve the latency and the efficiency of the DPD.

Additionally, we propose a different algorithm that selects the best parameters for the DPD regarding the current state of the PA. This algorithm uses a priori known PA characteristics, and finds the closest available characteristic to the current state of the PA. This approach relies on hypotheses tests and is agnostic to the channel noise and perturbation. Then, for each a priori known characteristic, we consider having an optimal set of DPD parameters, learned for each PA. Thus, the “selector” chooses the adapted set of parameters for the DPD to linearize the current state of the PA.

Both algorithms are evaluated regarding a PA modeling whose nonlinearities are shifting. We demonstrate successful ability of both algorithms to deal with time-varying effects with low-latency, justifying their use in real-world applications. Eventually, the technical novelties presented here have been disseminated in the following contributions.

[P1] **A. Falempin**, R. Zayani and J-B. Doré, “Device for linearizing a power amplifier of a communication system by digital predistortion,” Issued in July 05, 2022, US2107230.

[J1] **A. Falempin**, J-B. Doré, R. Zayani and E. Calvanese Strinati, “Low-Complexity Adaptive DPD: From online optimization to meta-learning,” IEEE Transactions on Broadcasting, 2022.

Chapter 5: Implementing the DPD on hardware using few resources. In this chapter, we study the implementation of the **DPD** proposed in chapter Section 3. To optimize the latency and energy efficiency of our **DPD**, we consider acting on the quantization of the developed **NNs**. Indeed, by default, **NNs** perform operations of number coded on 32 bits. This is problematic for embedded systems where the available computing resources are limited. Besides, such operations are power hungry; then again not adapted to embedded systems. Thus, we propose a low-bit quantization of our **NN** based **DPD**. Namely, we investigate the effect of the quantization on our **NNs** using a post-training quantization, *i.e.* quantizing the parameters and outputs of the **NNs**. Besides, low-bit quantization introduces a high level of noise that must be mitigated to avoid severe performance degradation of the system. Thus, to cope with this issue, we perform a quantization-aware training on our pre-trained **DPD**. This allows taking into account the quantization noise during a new training. Therefore, we optimize the parameters to mitigate the quantization noise. Thanks to this method, we can enable low-bit quantization for our proposed **DPD**.

In addition, we propose to translate our **DPD** on **FPGA**. Thus, we propose **FPGA** designs for each **NN** composing the **DPD**. By doing so, we can provide an estimation of the resource consumption regarding the chosen quantization scheme. Numerical simulations and synthesis show that low-bit quantization can be achieved without any performance loss compared to a 32 bits quantization scheme, and consumes very few resources on an **FPGA**. Eventually, the technical content presented here have been disseminated in the following contribution.

[C2] **A. Falempin**, J. Laurent, J-B. Doré, R. Zayani and E. Calvanese Strinati, “On the Performance of Quantized Neural Networks based Digital Predistortion for PA linearization in OFDM systems,” In Proc. IEEE Conference on Vehicular Technology (VTC2022-Fall), London, UK, 2022.

Extra-thesis work: Using AI solutions for demapping: Some others works have been disseminated, but are not presented in the chapters of this thesis. Readers may read two papers related to the design of a neural network based demapper for non-coherent **MIMO** sub-TeraHertz systems. The first paper [J3], –see Appendix B, studies different demapping approaches including a **NN**. The second one [C4], –see Appendix C presents the implementation of this specific **NN** on a high efficient Digital Signal Processing (**DSP**) unit. The latter work shows that high throughputs can be achieved with a very low power consumption.

[J3] S. Bicaïs, **A. Falempin**, J-B. Doré and V. Savin, “Design and Analysis of MIMO Systems Using Energy Detectors for Sub-THz Applications,” IEEE Transactions on Wireless Communications, 3678-3690, 2022.

[C4] **A. Falempin**, J. Schmitt, T.D. Nguyen and J-B. Doré, “Towards Implementation of Neural Networks for Non-Coherent Detection MIMO systems,” In Proc. IEEE Conference on Vehicular Technology (VTC2022-Fall), London, UK, 2022.

Digital Pre-Distortion for Power Amplifiers

Contents

2.1	Introduction	11
2.2	Communication system model	11
2.3	The Power Amplifier: Essential but problematic	12
2.3.1	An insight on nonlinearities	12
2.3.2	Power amplifier modeling	12
2.3.3	Performance metrics	16
2.3.4	Power efficiency	17
2.4	Digital Pre-Distortion: Linearizing a Power Amplifier	19
2.4.1	DPD modeling	19
2.4.2	Indirect Learning Architecture (ILA)	20
2.4.3	State-of-the-art approaches	20
2.5	Conclusion	21

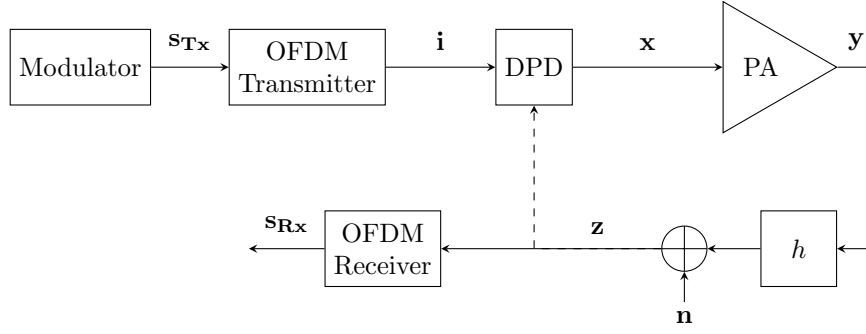


Figure 2.1: OFDM system with a DPD, a PA and a loopback link

2.1 Introduction

IN the first chapter of this thesis, we have seen that optimizing the energy efficiency of networks becomes crucial. Particularly, the Power Amplifier (PA), hardware component used for signal amplification, represents the most power hungry component of the Radio-Frequency (RF) transmission chain. It also presents nonlinear characteristics damaging the wireless signal transmission. Hence, researchers must find trade-offs between linearity and power efficiency to design power amplifiers. To improve this trade-off, linearization techniques have been investigated. In particular, Digital Pre-Distortion (DPD) is perceived as the most promising PA linearization technique [28]. It allows improving PA linearity, and permit to operate the power amplifier near its saturation region where its power efficiency is high, leading to a high global system energy efficiency. However, DPD can be difficult to perform and is subject to hardware implementation complexity. This chapter aims to provide an overview on the DPD technique to linearize a PA. We first introduce important elements about PA modeling and its issues when used in a modern wireless communication system. Then, we give insights on how DPD works and a literature review on this field.

The remainder of this chapter is organized as follows. Section 2.2 presents the considered communication system model used in this thesis. While Section 2.3 gives elements on PA modeling, Section 2.4 presents the DPD and typical solutions to perform it. Eventually, Section 2.5 draws some concluding remarks.

2.2 Communication system model

We consider the communication system model pictured in Figure 2.1 integrating a Quadrature Amplitude Modulation (QAM) scheme, an Orthogonal Frequency Division Multiplexing (OFDM) transmitter, a DPD and a PA. The received signal is prone to a complex perturbation coefficient h , e.g. phase impairment, and noise \mathbf{n} . The signal at the input of the receiver is then given by

$$\mathbf{z} = h\mathbf{y} + \mathbf{n}, \quad (2.1)$$

where $\mathbf{z}, \mathbf{y} \in \mathbb{C}^N$, $N = N_{OFDM} (N_{fft} + N_{CP})$, $h \in \mathbb{C}$ and $\mathbf{n} \sim \mathcal{CN}(0, \sigma^2)$ with σ^2 the noise variance. N_{fft} is the size of FFT used for the OFDM, N_{CP} denotes the length of the cyclic prefix, and N_{OFDM} the number of OFDM symbols.

In this chapter, we will consider no noise and no perturbation on the loop back link for clarity's sake.

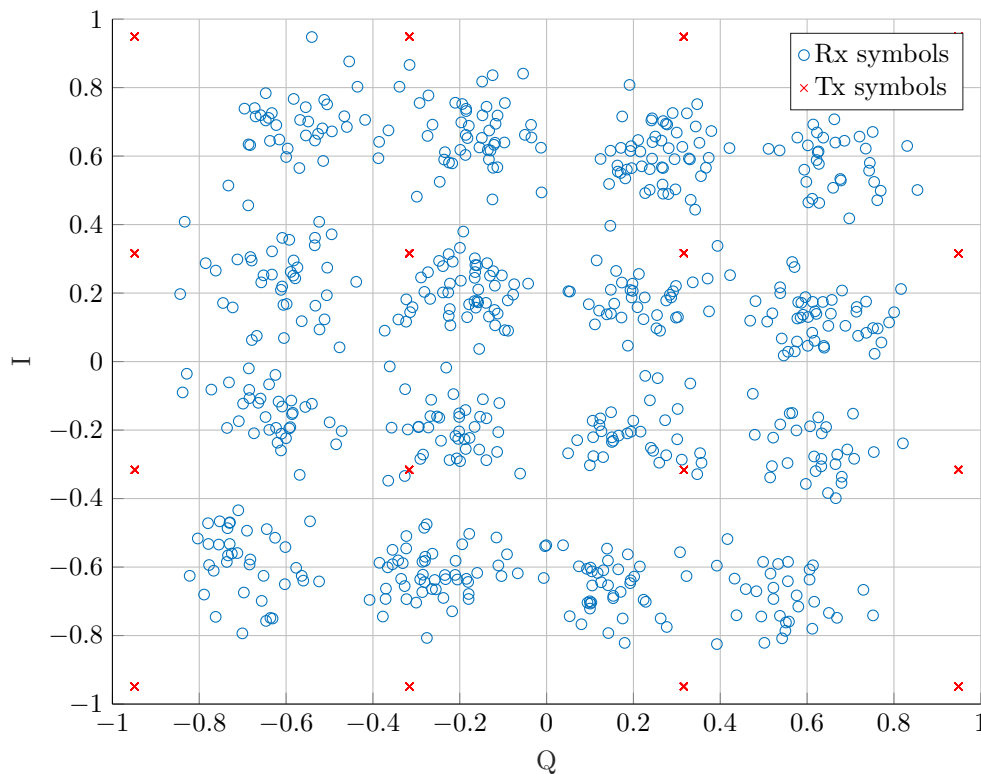


Figure 2.2: Received 16-QAM constellation using a PA

2.3 The Power Amplifier: Essential but problematic

A **PA** is a hardware component which converts a low magnitude signal into a high magnitude signal. Ideally, the **PA** should apply a gain to its input resulting in a linear amplified output. However, PAs are nonlinear because they contain transistors that present nonlinear behaviors. Admittedly, the **PA** will amplify the incoming signal, but distortions will also be injected resulting in in-band and out-of-band performance degradation.

2.3.1 An insight on nonlinearities

Let's consider a simple communication system composed of a **QAM** modulator, an **OFDM** transmitter and a **PA**. We consider a perfect channel without any noise to better show the impact of the **PA**.

We consider transmitting a 16-QAM constellation using **OFDM** with a 1024 **FFT** size. Figure 2.2 presents the received constellation after passing through the **PA**. One can observe that both amplitude and phase of the transmitted symbols have been altered. Due to the nonlinearities of the **PA**, it becomes impossible to retrieve the original symbols, leading to a poor transmission quality. Then, it becomes evident these nonlinearities shall be corrected.

2.3.2 Power amplifier modeling

The **PA** is the most power hungry component, and also a main source of nonlinearities in a communication system. Thus, designing **PA** models has been a focus for researchers in the past decades. In terms of modeling, there are two types of **PA** models: memoryless models and models with memory effects. If the **PA** frequency response exhibits flat characteristics over its entire working frequency range, the

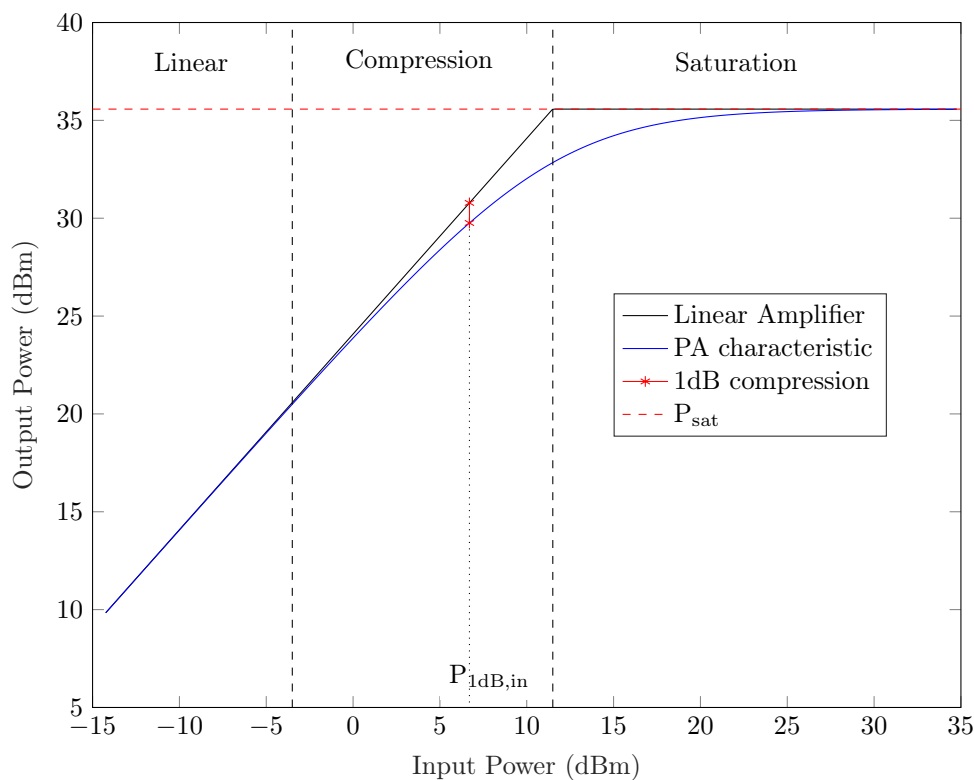


Figure 2.3: General characteristics of the power amplifier

nonlinearity is said to be frequency-independent or memoryless.

2.3.2.1 Power amplifier vocabulary

In this paragraph, we give some important keywords about the PA and its characteristics. A PA presents nonlinear transfer characteristics that will disturb the magnitude and phase of an incoming signal based on the input magnitude of the latter. Those characteristics are referred to as Amplitude-to-Amplitude (AM/AM) and Amplitude-to-Phase (AM/PM) transfer functions, characteristics or distortions. They are respectively denoted by f_ρ and f_ϕ .

On Figure 2.3, we present a typical AM/AM characteristic of a PA to illustrate the work regimes of the latter. The characteristic can be divided into three major regions:

- **Linear region** As its name indicates, the relationship between the input and the output is perfectly linear. Thus, the output corresponds to the input multiplied by the small gain of the PA.
- **Compression region** In this zone, the PA gain starts decreasing due to the nonlinearity. Therefore, we observe that the output is not proportional to the input anymore. The higher the input amplitude, the higher distortion in the compression zone. To evaluate the gain compression, we often measure the 1-dB compression point, $P_{1dB,in}$. As shown on Figure 2.3, $P_{1dB,in}$ represents the input magnitude where there is 1dB difference between the output magnitude and a linear amplifier. In other words, there is a 1dB decrease of the PA gain.
- **Saturation region** When the input magnitude becomes too high, the PA enters a saturation regime where the nonlinearity becomes severe. Eventually, the output magnitude becomes constant at a certain point.

Considering the system model in Figure 2.1, the output characteristics of the PA are given by,

$$|y| = f_\rho(|x|) \text{ and } \arg(y) = f_\Phi(|x|) + \arg(x) \quad (2.2)$$

where x and y are the PA input and output signals, respectively.

2.3.2.2 Memoryless models

Rapp model: The Rapp model [29] has become popular to theoretically design Solid-State Power Amplifiers (SSPA). It is a memoryless model presenting a nonlinear AM/AM characteristic given by

$$f_\rho(u) = \frac{Gu}{\left(1 + \left|\frac{Gu}{V_{sat}}\right|^{2p}\right)^{\frac{1}{2p}}}, \quad (2.3)$$

where u is the magnitude of the input signal. G represents the gain in the linear region and V_{sat} is the output saturation level. Finally, p rules the nonlinearity of the transfer function. As the p -value decreases, the function becomes more nonlinear and vice versa.

This model presents high AM/AM distortion but is linear on the phase which is unrealistic since all PAs exhibit AM/AM and AM/PM distortions. Thus, the regular Rapp model has been customized in [30] to take into account phase nonlinearities. It allows to mimic a realistic PA working with carriers frequencies above 6GHz even though memory effects are not taken into account.

Accordingly, the AM/AM characteristic is given by (2.3) and the AM/PM characteristic is given by

$$f_\Phi(u) = \frac{Au^q}{\left(1 + \left(\frac{u}{B}\right)^q\right)}, \quad (2.4)$$

where u is the magnitude of the input signal. A , B and q are fitting parameters to model the phase distortion.

Figure 2.4 presents the AM/AM and AM/PM characteristics with $p = 1.1$, $G = 16$, $V_{sat} = 1.9V$, $A = -345$, $B = 0.17$ and $q = 4$. The value of the parameters are chosen according to [30].

Saleh model: The Saleh model proposed in [31] is another popular way to design memoryless PAs. Similarly to the Rapp model presented above, it also presents AM/AM and AM/PM distortions that are modeled as follows:

$$\begin{aligned} f_\rho(u) &= \frac{\alpha_\rho u}{1 + \beta_\rho u^2}, \\ f_\Phi(u) &= \frac{\alpha_\Phi u^2}{1 + \beta_\Phi u^2}, \end{aligned} \quad (2.5)$$

where α_ρ , β_ρ , α_Φ and β_Φ are parameters to control the nonlinearities of the PA.

2.3.2.3 Models with memory

When the PA characteristic is frequency-dependent, it exhibits memory effects. In other words, the output of the PA depends on both current and past inputs. This kind of phenomenon can be observed while using wideband signals. Then, memoryless models become inaccurate to represent this phenomenon.

A popular way to model nonlinearities and memory effects of PAs is to use Volterra series [32]. If we consider the system model in Figure 2.1, in discrete time, the output y_n of the PA is given by

$$y_n = \sum_{k=1}^K y_{k,n}, \quad (2.6)$$

where

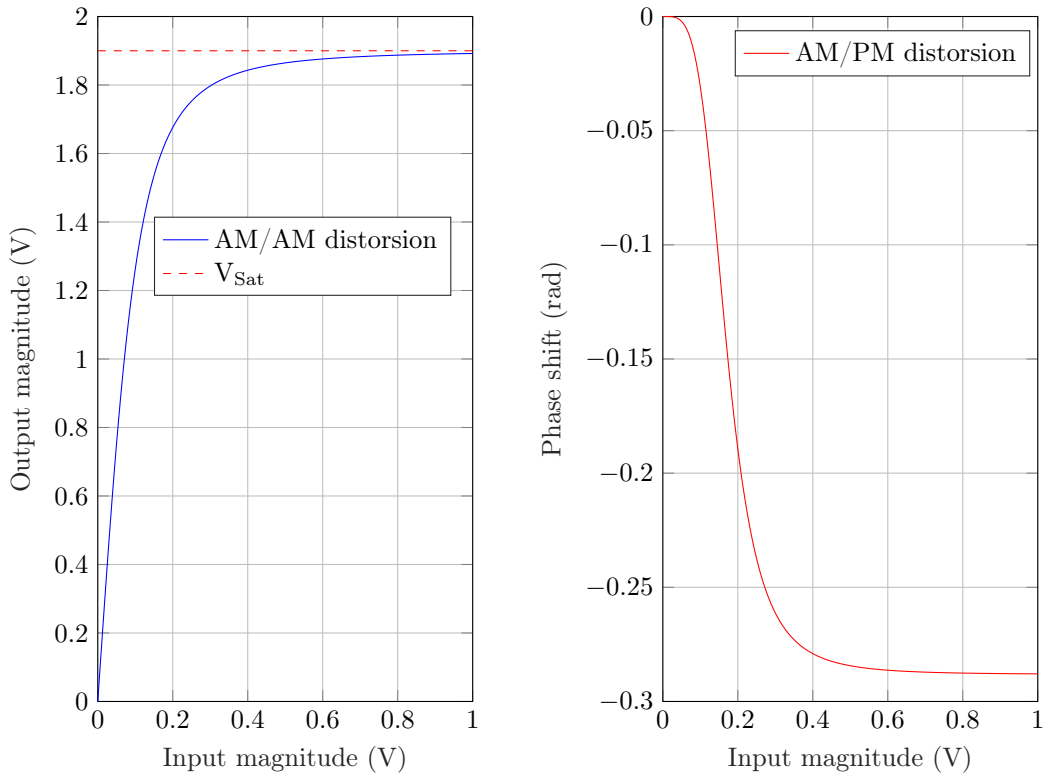


Figure 2.4: AM/AM and AM/PM characteristics of custom Rapp PA model

$$y_{k,n} = \sum_{i_1=0}^{M-1} \cdots \sum_{i_k=0}^{M-1} j_k(i_1, \dots, i_k) \prod_{j=1}^k x_{n,k-i_j}. \quad (2.7)$$

x_n denotes the discrete time input. $j_k(i_1, \dots, i_k)$ represents the Volterra kernel of order K and M denotes the memory depth.

Using these equations, we can model accurately all the nonlinearities of a PA. However, it is computationally expansive due to the memory depth and size of the kernel. Besides, there are other models allow representing PAs with memory effects, such as Wiener and Hammerstein models [33] that exhibit lower complexity than the Volterra series. Alternatively, one can also use memory polynomial model which is a truncation of the Volterra series [34]. The main advantage of this model is that it is less complex than the other ones while maintaining a good modelling of the memory effects.

2.3.2.4 Time-varying models

A PA can be subject to time-varying effects such as aging, temperature, electrical variations, *etc*. These effects are going to change the AM/AM and AM/PM characteristics of the PA. Among all effects affecting the PA, the time-varying ones are less considered in the literature. Still, in [35], authors consider a variation of the parameters defining the Saleh model. Here, to model a time-varying PA, we consider the Rapp model presented above. For instance, we consider the variation of the parameter p in equation (2.3). This parameter controls the linearity of the AM/AM characteristic. When p increases, the PA becomes more linear, and vice-versa. For our studies, we consider having $0.5 \leq p \leq 1.5$, leading to the gain variation presented on Figure 2.5.

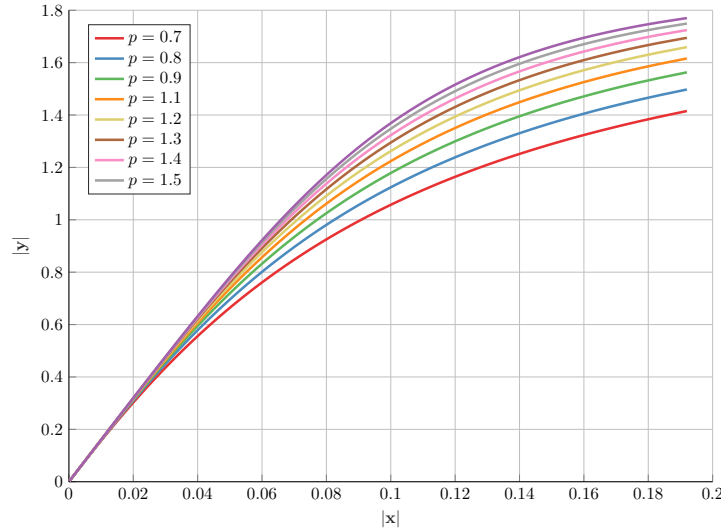


Figure 2.5: Variation of PA gain

2.3.3 Performance metrics

A signal passing through the **PA** may experience strong level of nonlinearities. To assess the degree of nonlinearities induced by the **PA**, we measure the performance of the communication system using several performance metrics. In this manuscript, we mainly focus on Error Vector Magnitude (**EVM**) and the Adjacent Channel Leakage Ratio (**ACLR**).

2.3.3.1 Error Vector Magnitude (EVM)

The **EVM** measures the error between the constellation sent by the transmitter and the one at the receiver, respectively denoted by \mathbf{s}_{Tx} and \mathbf{s}_{Rx} in Figure 2.1. Ideally, both constellations shall be identical. However, the transmission can suffer from perturbations such as channel noise, phase noise, nonlinearities due to the **RF** transmission chain. In our case, we want to evaluate the nonlinearities issued by the **PA**, so we consider that other noises are negligible compared to the **PA** nonlinearities. Then, the **EVM** is given by

$$EVM_{\%} = \sqrt{\frac{\mathbb{E}(|\mathbf{s}_{Rx} - \mathbf{s}_{Tx}|^2)}{\mathbb{E}(|\mathbf{s}_{Tx}|^2)}} \times 100, \quad (2.8)$$

$$EVM_{dB} = 20 \log_{10} \left(\frac{EVM_{\%}}{100} \right).$$

The lowest EVM, the best accuracy on the received constellation. An $EVM_{\%} = 0\%$ is the ideal case which is unfeasible in practice due to the perturbations induced in the transmitted signal.

In practice, the minimum **EVM** required for designing transmitters is standardized in [36] regarding the modulation scheme used, for Long Term Evolution (**LTE**). Table 2.1 summarize the **EVM** requirements which will serve as a reference for the performance evaluation of the **DPD** design presented later on.

2.3.3.2 Adjacent Channel Leakage Ratio (ACLR)

Due to its nonlinearities, a **PA** will induce Out-of-Band (**OOB**) spectral regrowth in the **OFDM PSD**. The **ACLR** is a figure of merit which evaluates the degree of **OOB** distortion. Figure 2.6 presents the **OFDM PSD** with $N_{fft} = 1024$ and 666 active subcarriers to reach a 10 MHz bandwidth with a 15 kHz subcarrier spacing. The main channel, *i.e.* the useful bandwidth, is colored in blue and first adjacent

Table 2.1: EVM requirements for LTE

Modulation scheme	EVM _%	EVM _{dB}
QPSK	17.5%	−15 dB
16-QAM	12.5%	−18 dB
64-QAM	8%	−22 dB

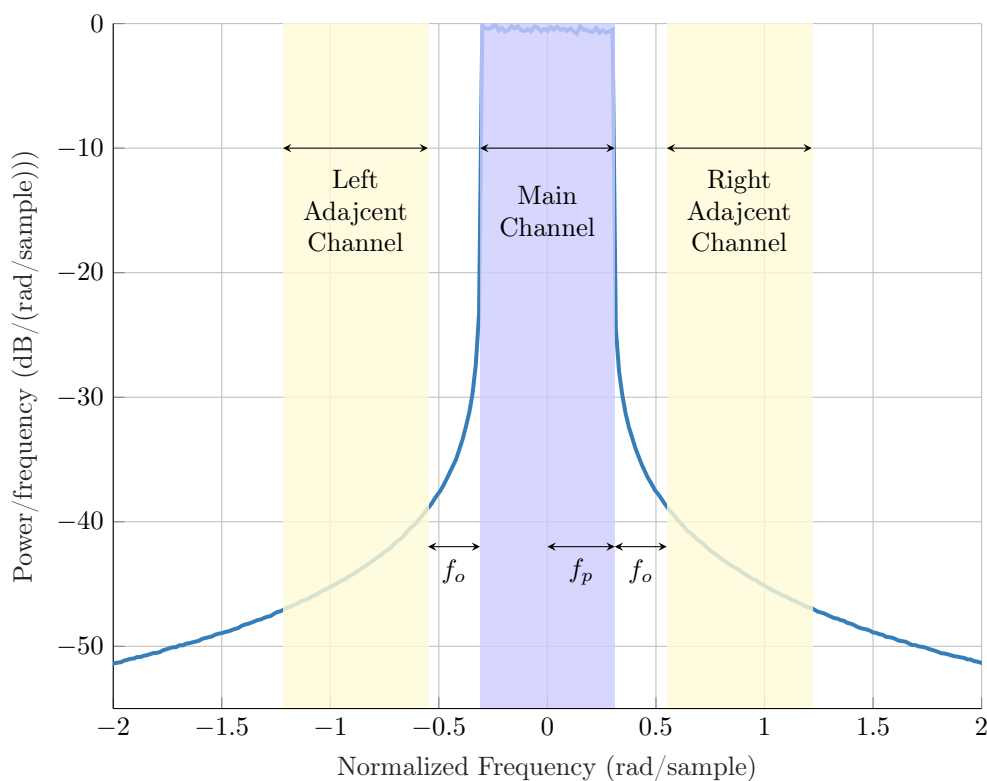


Figure 2.6: PSD of OFDM

channels are represented in yellow. The **ACLR** is the ratio of the power of the adjacent channels and the power of the main channel. Since the **OOB** distortion is not necessarily symmetric, we measure the **ACLR** in left and right adjacent channels, respectively denoted by $ACLR_{left}$ and $ACLR_{right}$. They are defined as

$$ACLR_{left} = \frac{\int_{-(2N+1)f_p+Nf_o}^{-(2N-1)f_p+Nf_o} PSD(i)}{\int_{-f_p}^{f_p} PSD(i)}, \quad (2.9)$$

$$ACLR_{right} = \frac{\int_{(2N+1)f_p+Nf_o}^{(2N-1)f_p+Nf_o} PSD(i)}{\int_{-f_p}^{f_p} PSD(i)},$$

with N the **ACLR** order. In our study, we only consider the first adjacent channels leading to $N = 1$.

2.3.4 Power efficiency

In the first chapter of this thesis, we have seen that **PA** is the most power-hungry hardware component at the base station. Then, it is important to ensure maximum power efficiency of the **PA**. One can

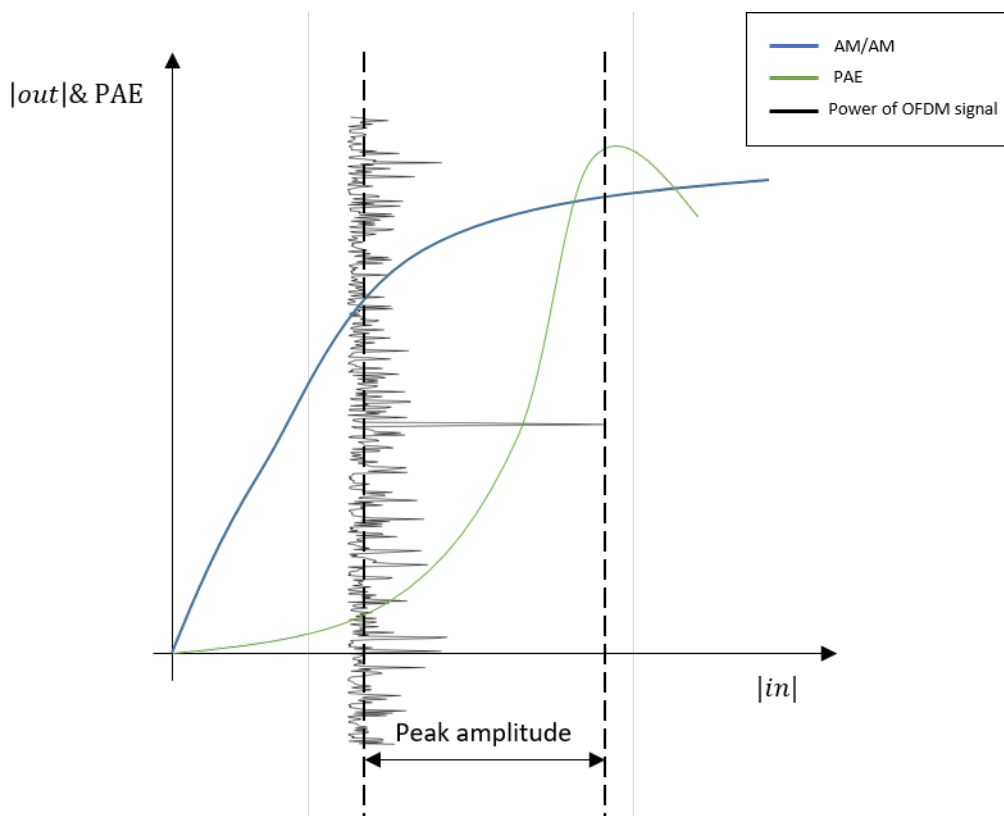


Figure 2.7: Impact of a high peak amplitude on the PA PAE

estimate the overall efficiency by expressing the Power Added Efficiency (PAE) of the PA given by

$$PAE = \frac{P_{out} - P_{in}}{P_{DC}}, \quad (2.10)$$

where P_{out} and P_{in} denote respectively the output and input powers of the PA. P_{DC} is the power consumed by the PA which can be estimated by measuring the current passing through the drain of the PA multiplied by the supplied DC voltage.

2.3.4.1 Trade-off between linearity and power efficiency

In wireless communications, we must ensure resiliency at the receiver side meaning that the PA shall not induce nonlinearities at the transmitter side. Ideally, input signals shall be in the linear zone of the PA to avoid errors at the receiver. As explained in Chapter 1, OFDM is the retained waveform for wireless communications systems thanks to its numerous advantages. However, OFDM exhibits a high PAPR. To avoid nonlinear distortions, we need to back off the signal and operate the PA in its linear region. Then, we define the Input Back-Off (IBO) as

$$IBO = \frac{P_{sat,in}}{P_{avg,in}}, \quad (2.11)$$

$$IBO_{dB} = 10 \log_{10} (IBO)$$

where $P_{sat,in}$ denotes the input signal power for which the PA saturates. $P_{avg,in}$ defines the input signal average power.

Figure 2.7 is a representation of the trade-off between linearity and power efficiency. On this figure, we can see the high PAPR exhibited by the OFDM signal. Besides, we can see that the PAE is maximum

near the saturation of the PA. One can note, that we must apply a large back off to avoid entering the saturation of the PA. Considering OFDM, to avoid saturation, we must apply a back-off to the signal leading to $IBO_{dB} = 12\text{dB}$.

Besides, the presented PA is highly nonlinear, especially on the AM/PM distortion. To have a system perfectly linear, one must have a very large back-off which induces a poor PAE leading to an inefficient transceiver. Thus, one must find techniques to ensure PA linearity while having the highest PAE. As stated in the introduction of this thesis, two complementary techniques have been studied, namely PAPR reduction [15] and PA linearization [20]. Both techniques allow linearizing the PA while maintaining the highest power efficiency. In this thesis, we will specifically focus on PA linearization using DPD.

2.4 Digital Pre-Distortion: Linearizing a Power Amplifier

DPD is a PA linearization technique which has been widely studied in the literature [28, 37] and has been shown to be the most promising PA linearization technique. It consists in finding the inverse characteristic of the PA and apply it before the latter so that the resulting system DPD and PA is linear. Hereafter, we give some insight about DPD and how it is performed in the literature.

2.4.1 DPD modeling

The goal of the DPD is to apply a distortion to the OFDM signal such that when it is amplified by the PA, the resulting system DPD and PA is linear. To do so, one must find the inverse characteristic of the PA. This is equivalent to

$$H_{PA} \circ H_{PD} = I, \quad (2.12)$$

where H_{PD} is the transfer function of the DPD function, H_{PA} the transfer function of the PA, and I the identity function.

Theorem 1. *Let $f: C \rightarrow D$ be a function. f admits an inverse, $f^{-1}: D \rightarrow C$, if and only if f is bijective. Then, we have*

$$f^{-1}(d) = c \Leftrightarrow d = f(c). \quad (2.13)$$

Corollary 1. *Let $f: C \rightarrow D$ be a bijective function. It follows that*

$$f^{-1} \circ f = I_C, \quad \text{and} \quad f \circ f^{-1} = I_D, \quad (2.14)$$

where I_C and I_D denote the identity function on C and D , respectively.

Thus, to find the ideal DPD of the PA, the latter must have a bijective transfer function to derive its DPD function. From equation (2.14), problem (2.12) becomes

$$H_{PA} \circ H_{PD}^{-1} = I. \quad (2.15)$$

As a toy example, we consider a PA exhibiting only AM/AM distortion. Figure 2.8 presents the DPD function that will linearize the PA. As we can see, the goal is to obtain a linearized amplifier. Thus, the challenge is to derive the DPD characteristic.

Remark 1. *If the PA characteristic is not bijective, we can consider a piecewise linearization. Doing so, we linearize bijective functions. Then, one must also represent the PA characteristic into a succession of bijective functions.*

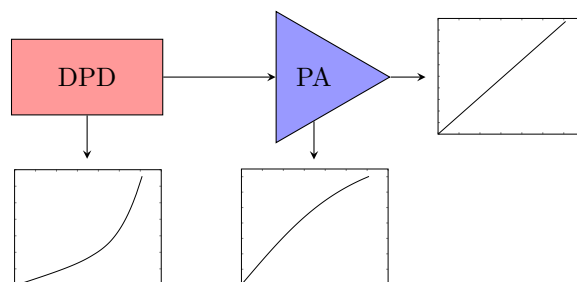


Figure 2.8: Transfer function of DPD, PA and resulting system

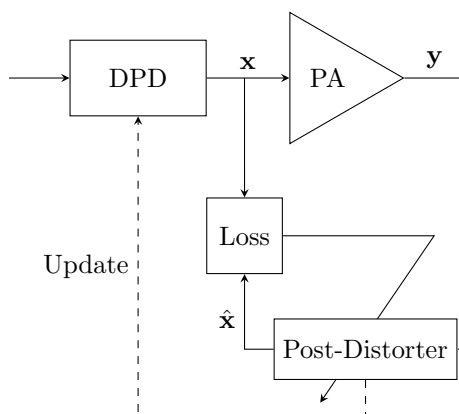


Figure 2.9: Indirect Learning Architecture

2.4.2 Indirect Learning Architecture (ILA)

To derive the **DPD** module, it exists two approaches named Direct Learning Architecture (**DLA**) and Indirect Learning Architecture (**ILA**). We will only consider the **ILA** to derive the **DPD**. The approach is presented on Figure 2.9. It consists in finding a post-inverse of the **PA** using a parameter estimation technique described later. Besides, to use the post-distorter as a pre-distorter, the **PA** characteristic must be bijective according to (2.13).

2.4.3 State-of-the-art approaches

Performing **DPD** has been studied and exists since many years. It mainly exists four categories of pre-distorters: Look-up Tables (**LUTs**), Volterra, and polynomial methods, and deep learning solutions. We give insights on each approach.

Look-up Tables (LUTs): These are arrays that map input values to output values, thus describing a discretized mathematical function. **LUTs** are often used to estimate a pre-distortion function [38]. In the later work, authors did a full study on usage of **LUTs** to derive the **DPD** function of a **PA**. The general idea is to discretize the **PA** **AM/AM** and **AM/PM** characteristics and then approximate the linear response to know the gain of the **PA**. To derive the inverse function, we have to find a value which when passing through the **PA**, the result becomes linear. Thus, **LUTs** are used to store the **PA** characteristic and then an algorithm finds the compensation to linearize it [39]. This method is widely used to estimate the **DPD** function. However, it remains mainly static and can exhibit high complexity if you want a good estimation of the **DPD**.

Volterra series: Similarly to **PA** modelling, one can consider using Volterra series to model the **DPD**, – refer to Section 2.3.2.3. Then, one must find the value of the Volterra kernels. Estimating those values can be achieved using the least square algorithm [40]. This is computationally intensive since the

Volterra model may exhibit a high number of kernels, and may present high memory depth.

Polynomial estimation: It is another technique which is based on a polynomial design of both **PA** and **DPD** characteristics. The polynomial design is often conducted using truncated Volterra series that can even model memory effects. Or else, a simple model proposed in [41]. Thus, to find the pre-distorter characteristic, one must find the inverse of a n -th order polynomial. Inverting this kind of polynomial has been introduced in [42]. This method can be challenging to implement due to the high complexity of the inverting algorithm. Other algorithms use least squares filters which are also complex and slow to converge. Therefore, they are not suited for real-time applications where latency and energy efficiency are primordial.

Neural networks: Neural Networks (**NNs**) are a machine learning technique that learn how to solve a problem by using data. They are widely used to solve nonlinear problems. Then, the use of such technique has been considered to mainly help designing the **DPD**, and estimates its parameters. In essence, the complexity of such technique is better than using the Volterra series because fewer parameters are needed to compute the **DPD**. For instance, in [43], the authors use a **NN** to find the **DPD**. From the output and input signals of the **PA**, they showed that an accurate **DPD** can be derived. Still, the complexity of such technique remains high because of the “black-box” paradigm concerning the **NN** architecture design. In the next chapters, we consider using **NNs** to propose a low-complexity **DPD** that can deal with time-varying **PAs**.

2.5 Conclusion

In this chapter, we gave some practical elements about **PA** modeling and more specifically about the nonlinear behavior of this component. The latter exhibits high nonlinearities that modify the amplitude and phase of the amplified signal. Those nonlinearities are referred to as **AM/AM** and **AM/PM** distortions. We use **EVM** and **ACLR** to quantify the level of nonlinearities, which measure in-band and **OOB** distortions induced by the **PA**. To avoid nonlinearities, backing off the input signal is often considered but results in poor efficiency of the **PA**. Thus, linearizing techniques have emerged to compensate the distortions induced by this hardware component. The most promising technique called **DPD** has proven its efficiency to linearize the **PA** and operate it at lower **IBO**. However, deriving a **DPD** module can be challenging due to the high-degree of nonlinearity. Some classical techniques such as **LUTs** and polynomial design have been considered but still exhibit high hardware complexity and are mainly static, *i.e.* the **PA** characteristics are fixed in time. In the next chapters, we introduce a **DPD** function based on Artificial Intelligence (**AI**) techniques to tackle down hardware complexity and time-varying aspects of the **PA**.

Part II

Efficient AI-based Digital Pre-distortion solutions for PA Linearization

Enabling Low-Complexity Neural Networks for Digital Pre-Distortion

Contents

3.1 Introduction	23
3.1.1 Motivations	24
3.1.2 Related Work	24
3.1.3 Contributions	24
3.2 Background on Deep Learning	25
3.2.1 Theory	25
3.2.2 Toy example: the sine function	28
3.3 Deep Learning for Digital Pre-Distortion	30
3.3.1 State-of-the-art approach: Cartesian DPD	30
3.3.2 Towards a Low-Complexity DPD Neural Network (LCDPDNN)	30
3.3.3 Numerical simulations	33
3.4 Conclusion	40

3.1 Introduction

To break down the complexity and ensure performance of the RF wireless transmission chain, our approach focuses on the design of NNs to perform DPD for PAs. In this chapter, we first give a background on deep learning and its usage for pre-distortion. Then, we propose a custom design of NNs to achieve low-complexity architecture with improved performance regarding state-of-the-art solutions. The proposed solution is dedicated to the problem to solve and allows to escape from the “black-box” paradigm in NNs.

3.1.1 Motivations

Since the emergence of deep learning techniques, e.g. **NNs**, many solutions have been proposed to try solving problems related to wireless communication, as shown in the first chapter of this manuscript. Regarding the conception of **DPD**, one can see in the previous chapter that classical solutions, namely **LUTs** and polynomial methods, exhibit a high level of complexity to achieve near-optimality in terms of performance. This is undesirable for systems that rely on real-time considerations and cost-effective design. Many works on **DPD** based on **NNs** have been conducted, but the problem of complexity is often neglected for the benefit of performance. To tackle this issue, our approach mainly focuses on finding the best architecture design to perform **DPD** with **NNs**. This requires to go beyond the “black-box” paradigm, further explained, where the architecture is a succession of layers performing matrix multiplications. This is the main reason of high complexity **NN** designs. Then, our goal is to propose a non “black-box” design to achieve low-complexity solution.

3.1.2 Related Work

As stated in the previous chapter, **DPD** has been widely investigated the last decades. With the breakthrough of machine learning, multiple works have emerged to propose **DPD** based on **NNs**. In [43], authors have proposed a **NN** based **DPD** using Cartesian information of input and output signals of the **PA**. The solution performs well but is not designed to achieve low-complexity since they employ a regular Fully Connected Network (**FCN**) design. Similarly, the work proposed in [44] designs classic **FCN** architectures using polar information of the **OFDM** signal to perform **DPD**. Performance is improved compared to the first solution, but complexity is still challenging here. More recently, a new design has been proposed in [45] to avoid using **ILA** and thus being more robust to noise. However, this proposal needs two **NNs**, one for modeling the **PA** and another one to derive the pre-distortion function. Again, this leads to increased complexity even though the solution allows to correct all nonlinearities of the chosen **PA**. Finally, a sophisticated solution is proposed in [46] which linearize **PAs** with memory effects. The performance of this solution is good, but complexity is really high and hardware implementation of it would require high computing resources.

We have seen that most solutions proposing **NN**-aided **DPD** exhibit high complexity. This is not suited to real-time applications where latency and throughput are important. Moreover, high complexity also means increased usage of computing resources and by extension less energy efficient.

3.1.3 Contributions

The contributions of this chapter are:

- **Polar decomposition for DPD:** we choose to tackle **PA** distortions separately using a polar decomposition of the input and output signals of the **PA**. Many works only focus on Cartesian approach of the **DPD**, except from the work in [44]. The latter shows same performance as a Cartesian representation of the signals. Using our approach, we demonstrate that the polar decomposition leads to better performance than the Cartesian one with less computational complexity.
- **Custom NN architecture for AM/AM pre-distortion:** we propose a custom **NN** architecture specifically design to tackle the **AM/AM** distortion using input and output signals amplitudes of the **PA**. This architecture permits achieving low-complexity compensation of the **AM/AM** distortion.
- **Custom NN architecture for AM/PM pre-distortion:** similarly to the above proposal, we also design a specific **NN** architecture to cope with **AM/PM** distortion using input signal amplitude of the **PA** and the phase shift induced by the latter. This design allows reaching low-complexity of the pre-distortion estimation to linearize the **AM/PM** effects.

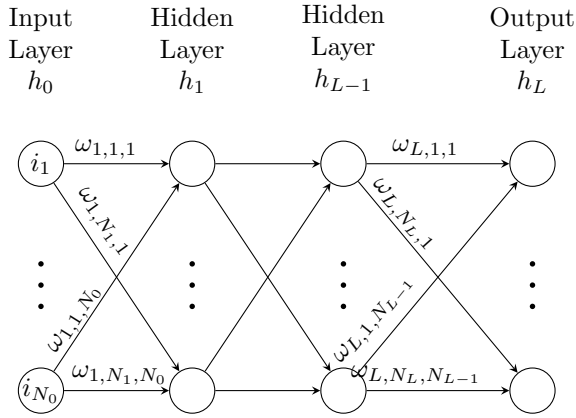


Figure 3.1: Generic NN architecture

The technical content is based on [47] and [48] in part. The remainder of this chapter is organized as follows. Section 3.2 gives a background on deep learning and how to apply it to a simple problem. Section 3.3 presents a common state-of-the-art approach to perform **DPD** using a **NN**. We compare this solution to our main contribution which reduces the global complexity of the **DPD** module. Numerical results are also presented in this section.

3.2 Background on Deep Learning

3.2.1 Theory

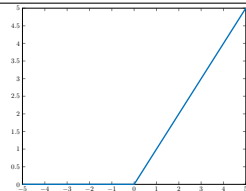
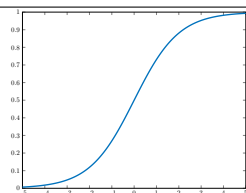
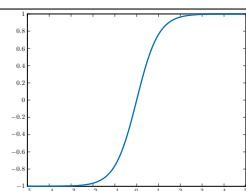
We introduce here some background about deep learning. Some generalities have been covered in the first chapter of this manuscript. Deep learning is a machine learning technique relying on the use of **NNs** which are inspired by human brain neurons and synapses. The essence of deep learning is to find a transfer function between single/multiple inputs and single/multiple outputs using a **NN**.

The process of deep learning is the same as machine learning and can be summarized as follows:

- *Data*: It is the starting point of every machine learning problem. Basically, we have to process high quality and sufficient data in order to produce a good estimation of the function needed.
- *Architecture design*: From the problem type (regression, classification, clustering, etc), data and some analysis on it, we design an architecture using **NNs** that will be used to map a relation between the input data and the desired output.
- *Learning*: Once we have our data, we teach our **NN** architecture how to find the function map the input data to the desired output. This is done using a learning phase which will calibrate the parameters of the **NN**.
- *Inference*: Finally, when the learning phase is successful, the **NN** now models correctly the transfer function between the input and desired output. Thus, it can be used to make predictions regarding its input.

Data: In every machine learning problem and especially deep learning ones, data is primordial. Indeed, a large amount of data is required to learn the transfer function between input and output. To solve any problem, we often give large datasets to **NNs** without considering the aspect of data consumption because we want the best performance. The main challenge about data is to find the sufficient dataset to have an efficient learning. Usually, this can be found empirically or by adopting custom **NN** designs

Table 3.1: Common activation functions

Activation	Formula	Illustration
ReLU	$\max(0, x)$	
sigmoid	$\frac{1}{1+e^{-x}}$	
tanh	$\frac{e^x - e^{-x}}{e^x + e^{-x}}$	

and/or learning strategies.

Architecture Design: NNs are often represented as in Figure 3.1. This is a FCN architecture also called multilayer perceptrons in the literature. This kind of NN is widely used because it is easy to use and can approximate whatever continuous function as stated in [49]. Thus, we have three components in this architecture:

- *Input layer:* it represents the input data, named features, that is needed to solve the desired problem.
- *Hidden layers:* it is the logic core of the NN. Each hidden layer h_l with $l = 1, \dots, L$ is composed of N_l neurons that are performing nonlinear operations. A neuron is performing a weighted sum of its inputs, adds a bias and apply a function called “activation”. Thus, the output of the k -th neuron from hidden layer l is given by,

$$o_{l,k} = f_l \left(\sum_{j=1}^{N_l} \omega_{l,k,j} o_{l-1,j} + b_{l,k} \right), \quad (3.1)$$

where l is the index of the current layer, $l = 0$ corresponds to the input layer. j denotes the index of the neuron of the previous layer. Thus, $\omega_{l,k,j}$ are the weights multiplying the outputs of the previous layer for the current neuron k of layer l . Besides, $b_{l,k}$ is the bias term added by the neuron k of layer l . Finally, the function f_l represents the neuron activation function in layer l . Table 3.1 presents some usual activation functions commonly used.

Remark 2. For clarity's sake, we will consider that the parameters $\omega_{l,k,j}$ and $b_{l,k}$ are encapsulated in a set of weights denoted by θ .

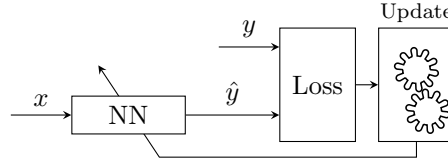


Figure 3.2: Learning process of a NN

Remark 3. We suppose that the neurons belonging to the same layer use the same activation function for simplicity. In practice, having different activation functions per layer will not help to learn the pattern between input and desired output.

- **Output layer:** it is the final layer of the **NN** which produces the desired output. It is composed of N_L output neurons corresponding to the output dimension we want.

Remark 4. The input data are organized in a matrix with each column representing an input. Each input marked $i_1 \dots i_{N_0}$ do not perform any operations. The output layer is the last layer of the **NN** and also performs operations regarding the problem type to solve.

Remark 5. The **NN** architecture presented in Figure 3.1 is a generic **FCN** architecture and does not represent all the existing designs. This architecture refers to the concept of “black-box” design which often exhibit high complexity.

Important note 1. For the rest of this thesis, we will consider that $FCN(N_0, N_1, N_2)$ denotes a **FCN** neural network with N_0 inputs, one hidden layer composed of N_1 neurons and N_2 outputs.

Learning: Once the architecture design has been conducted, a learning phase must be done in order to calibrate the parameters of the **NN**. Specifically, the parameters $\omega_{l,k,j}$ and $b_{l,k}$ are optimized during this stage. It exists two types of learning, supervised and unsupervised. Let $g : E \mapsto F$ be an arbitrary function with E and F being sets. We have $y = g(x)$ with x and y being the input and output respectively. To perform supervised learning, we give x and y to the **NN** so that it learns the function g . Regarding unsupervised learning, we only give x to the **NN**, and we let it produce the output, e.g. clustering algorithms. In our case and for the rest of this manuscript, we will only consider supervised learning.

Figure 3.2 represents the learning process of a **NN**. The input x goes forward to the **NN** which produces a value \hat{y} . Then, using a loss function, e.g. Mean Squared Error (**MSE**), the produced value \hat{y} is compared to the expected value y . Finally, we use a learning algorithm to optimize the loss function and update the parameters of the **NN**. Most learning algorithms for **NNs** are based on gradient descent algorithms because they provide excellent optimization performance for machine learning tasks [50]. Typically, a gradient descent based algorithm will update the parameters of the **NN** by calculating gradients of each parameter w.r.t. to the loss. The gradients are calculated by backpropagating the errors through the **NN** using the backpropagation algorithm [51]. The update rule of the **NN** parameters is given by,

$$\theta^i = \theta^{i-1} - \eta \nabla_{\theta^{i-1}} \mathcal{L}(\theta^{i-1}, \mathcal{D}), \quad (3.2)$$

where θ defines the weights of the **NN**, i.e. $\omega_{l,k,j}$ and $b_{l,k}$. i represents the gradient step. We suppose that at step $i = 0$, θ^0 is initialized randomly using a specific distribution. $\mathcal{L}(\cdot, \cdot)$ defines the loss function between the dataset \mathcal{D} and the values produced by the **NN** using the weights θ^{i-1} . Eventually, the

Table 3.2: Hyperparameters for sine fitting

Parameter	Symbol	Value
Learning rate	η	10^{-3}
Number of neurons	N_1	25
Batch size	B_s	32

operator ∇ defines the gradient operator and η the learning rate which controls the speed of the gradient descent.

3.2.2 Toy example: the sine function

We give here a simple example to apply the aforementioned deep learning process. Let define $y = \sin(x)$, $x \in [-\pi, \pi]$. We want to design a **NN** which learns the sine function within a specific range. We are trying to solve a regression problem.

Data: We construct a dataset $\mathcal{D}(\mathbf{x}, \mathbf{y})$ representing our sine function. This dataset will be fed to the **NN** during the learning stage.

Architecture Design: We propose a simple design based on the generic architecture presented in Figure 3.1. For this case, the input layer only contains \mathbf{x} . The output layer will have only one neuron performing a linear operation to produce $\hat{\mathbf{y}}$. Then, we use one hidden layer with $N_1 \geq 25$ neurons. The number of neurons is determined empirically or can be tuned using hyperparameters optimization techniques [52]. Concerning the number of hidden layers, it has been shown that using only one hidden layer, a **NN** can approximate whatever continuous function within a specific range [49].

Each neuron performs a Rectified Linear Unit (**ReLU**) activation – see Table 3.1. **ReLU** is often used because it is easy to compute and less prone to vanishing gradient problems during learning.

Learning: Learning stage is conducted using the gradient descent based algorithm presented in (3.2) to tune optimize a **MSE** loss function. To improve the convergence of this algorithm, it is common to use a batched version of it, called Stochastic Gradient Descent (**SGD**). It performs the learning on small subsets of data called batches. Table 3.2 presents the hyperparameters chosen to perform the learning of the sine function. Those parameters are chosen empirically. B_s denotes the size of each batch used for the **SGD** algorithm.

Inference: Once the learning stage is completed, one can use the **NN** to perform predictions. Figure 3.3 shows the actual sine function in blue and the predicted one in red. One can clearly see that both representations are identical. We can still notice that the **NN** is less accurate when the amplitude is maximum. This issue can be corrected by increasing the number of neurons or using another architecture.

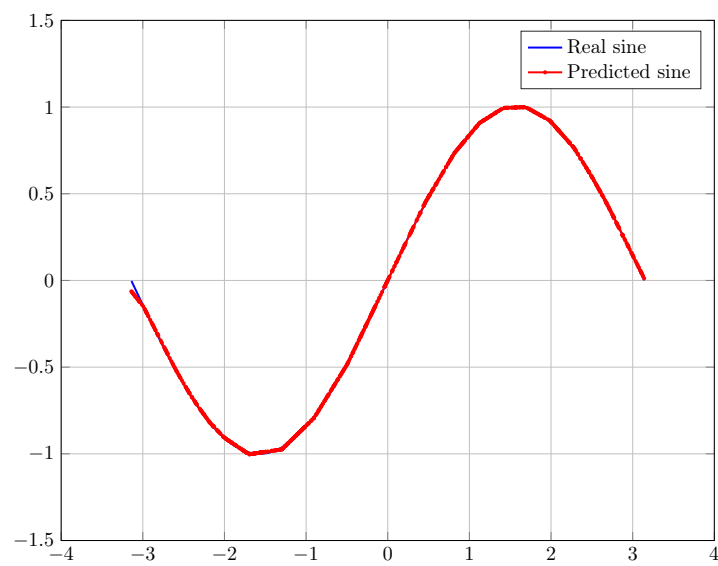


Figure 3.3: Fitting of sine function

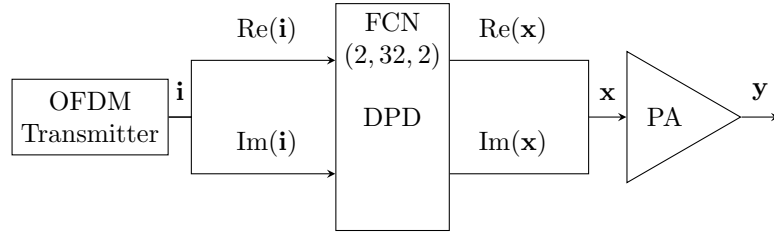


Figure 3.4: Classic NN based DPD

3.3 Deep Learning for Digital Pre-Distortion

First, we present here a classical state-of-the art approach to perform **DPD** based on **NNs**. We then put forward our solution whose designs allow achieving low-complexity implementation. The system model used is presented on Figure 2.1, in Chapter 2, with perfect return link and no noise.

3.3.1 State-of-the-art approach: Cartesian DPD

Architecture Design: A common design of **NN** based **DPD** has been proposed in [43] and reused in part in [45]. The architecture of this approach is presented on Figure 3.4. As one can see, the signal coming from the **OFDM** transmitter passes through a generic **FCN** architecture composed of two inputs, one hidden layer of 32 neurons and two outputs. Since **NNs** cannot perform operations on complex numbers directly, a workaround is to separate real and imaginary parts as two different inputs and outputs.

Data: As we can see on Figure 3.4, the **NN** uses Cartesian representation of the signal. It means that to perform learning, one needs a dataset composed of real and imaginary parts of \mathbf{x} and \mathbf{y} . Hence, the required data will be encapsulated in the dataset $\mathcal{D}_{IQ} ([\text{Re}(\mathbf{y}); \text{Im}(\mathbf{y})], [\text{Re}(\mathbf{x}); \text{Im}(\mathbf{x})])$.

Learning: Learning is conducted using **ILA** – see Figure 2.9 – and we optimize a **MSE** loss function \mathcal{L} using Gradient Descent (**GD**) algorithm (3.2) *w.r.t.* \mathcal{D}_{IQ} . Thus, this kind of architecture learns how to correct both **AM/AM** and **AM/PM**.

Besides, the main issue of this solution is its architecture which still exhibits high complexity due to the “blackbox” design of the **NN**. This leads to potential issues when used for real-time applications due to the computational latency.

3.3.2 Towards a Low-Complexity DPD Neural Network (LCDPDNN)

We present here our main contribution to break down the hardware complexity of **NN** usage for **DPD**. The final goal is to reduce the latency and improve throughput of real-time systems using **DPD**. To achieve this purpose, we propose a custom design putting forward a low-complexity **NN** architecture contrary to state-of-the-art solutions. This custom design relies on tackling separately **AM/AM** and **AM/PM** distortions induced by the **PA** using a polar decomposition. We show in simulations that, using specific **NN** design and polar decomposition rather than Cartesian representation, allows achieving better results.

Architecture Design: The Low-Complexity DPD Neural Network (**LCDPDNN**) is composed of two neural networks. Each neural network represents a function correcting respectively the **AM/AM** and **AM/PM** distortions. The architecture of both neural networks is presented in Figure 3.5. It must be emphasized that these neural networks respectively use amplitude and phase information of the signal. Moreover, the design is conducted in a goal-oriented way, meaning that we specifically design these

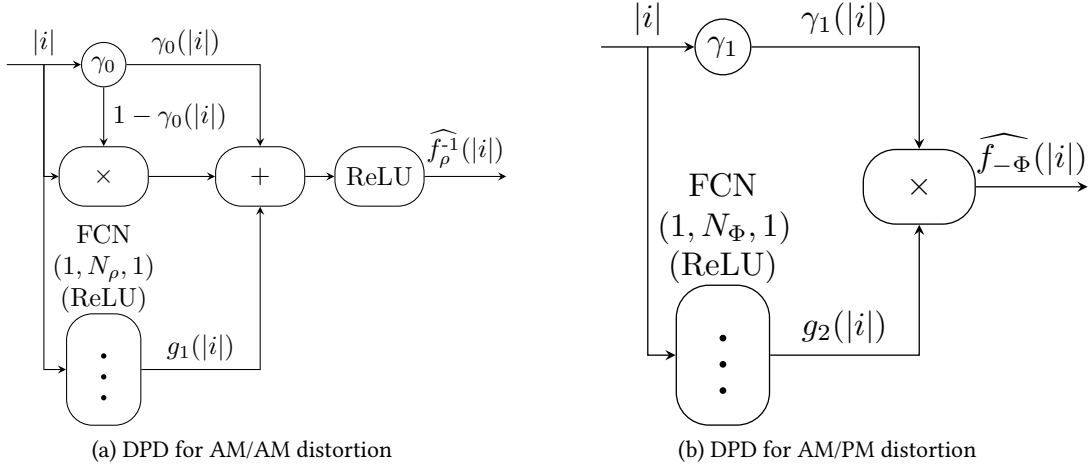


Figure 3.5: Architecture of the LCDPDNN for AM/AM and AM/PM distortions

NNs to deal with nonlinear issues caused by the PA. This allows to enable low-complexity contrary to “blackbox” design.

3.3.2.1 Neural Network for AM/AM DPD

In this paragraph, we present the architecture of the neural network allowing to inverse the **PA AM/AM** characteristic. The architecture is represented on Figure 3.5a. First, it can be noticed that the latter is fully customized in order to resolve the specific issue of inverting the **PA AM/AM** characteristic. The choice of such architecture relies on the shape of **AM/AM** characteristics. To lower the complexity, we propose using a single sigmoid neuron to estimate the inverse function. In addition, a layer of **ReLU** functions is used to improve the estimation produced by the sigmoid. Based on the system model presented in Section 2.2, the NN presented here takes the output amplitude of the PA and predicts the input amplitude. This NN is optimized to approximate the function \widehat{f}_ρ^{-1} such that,

$$(f_\rho \circ \widehat{f}_\rho^{-1})(|i|) = G|i|. \quad (3.3)$$

Important note 2. According to corollary (2.14), f_ρ must be bijective for efficient linearization of the **PA**.

Looking at Figure 3.5a, we define

$$\begin{aligned} \widehat{f}_\rho^{-1}(u) &= \text{ReLU}(g_0(u) + g_1(u)), \\ \text{where } \begin{cases} g_0(u) &= \gamma_0(u) + (1 - \gamma_0(u))u, \\ \gamma_0(u) &= \left(1 + e^{-\alpha(u - \omega_\rho)}\right)^{-1}, \\ g_1(u) &= \sum_{j=1}^{N_\rho} \omega_j^\rho [\text{ReLU}(\mathbf{W}_\rho (u - \omega_\rho) + \mathbf{b}_\rho)]_j + \beta_\rho, \end{cases} \end{aligned} \quad (3.4)$$

where $\alpha, \omega_\rho, \omega_j^\rho, \beta_\rho \in \mathbb{R}$, $\mathbf{W}_\rho \in \mathbb{R}^{1 \times N_\rho}$ and $\mathbf{b}_\rho \in \mathbb{R}^{1 \times N_\rho}$ are optimized during the learning phase. N_ρ denotes the number of neurons. The function g_0 allows to model the inverse of the **AM/AM** characteristic. Specifically, γ_0 models the correction of the nonlinearities. The second term of g_0 permits applying γ_0 after the linear part of the **PA**. Thus, g_0 allows to correct the nonlinearities induced by the

AM/AM distortion while perfectly conserving the linear zone. Finally, the g_1 function is instrumental in refining the correction brought by g_0 . It is acting as a fine-tuning for the DPD.

3.3.2.2 Neural Network for AM/PM DPD

We present here the **NN** architecture used to correct phase distortion due to the **PA**. The **NN** takes the input amplitude and predict the opposite of the phase shift. The architecture is different from the **NN** dedicated to find the inverse of the **AM/AM** characteristic because here we need to estimate the non-linear phase distortion and take its opposite. Then, fewer operations are required resulting in lower complexity. This **NN** is optimized to find the function $\hat{f}_{-\Phi}$ such as,

$$f_{\Phi}(|i|) + \hat{f}_{-\Phi}(|i|) = 0. \quad (3.5)$$

Looking at Figure 3.5b, we define

$$\begin{aligned} \hat{f}_{-\Phi}(u) &= \gamma_1(u)g_2(u), \\ \text{where } \begin{cases} \gamma_1(u) = \left(1 + e^{-\beta(u-\omega_{\Phi})}\right)^{-1}, \\ g_2(u) = \sum_{j=1}^{N_{\Phi}} \omega_j^{\Phi} \text{ReLU}[(\mathbf{W}_{\Phi}(u - \omega_{\Phi}) + \mathbf{b}_{\Phi})]_j, \end{cases} \end{aligned} \quad (3.6)$$

where $\beta, \omega_{\Phi}, \omega_j^{\Phi} \in \mathbb{R}$, $\mathbf{W}_{\Phi} \in \mathbb{R}^{1 \times N_{\Phi}}$ and $\mathbf{b}_{\Phi} \in \mathbb{R}^{1 \times N_{\Phi}}$ are optimized during the training phase. N_n^{Φ} denotes the number of neurons. To design the **NN** performing the operation (3.6), we analyze the phase shift distortion induced by the **PA**. The phase shift is null in the linear zone and can be either negative or positive after the linear zone. Thus, using a sigmoid multiplied by a weighted sum of **ReLU** functions allows to respect the linear zone and correct the nonlinearities.

Data: Our designed **NNs** work with polar information of the signals \mathbf{x} and \mathbf{z} . This allows proposing a low-complexity design as shown above. Thus, we build two datasets, $\mathcal{D}_{\rho}(|\mathbf{z}|, |\mathbf{x}|)$ and $\mathcal{D}_{\Phi}(|\mathbf{x}|, f_{\Phi}(\mathbf{z}))$ for training respectively **AM/AM** and **AM/PM DPD NNs**. As a reminder, f_{Φ} denotes the phase shift induced by the **PA**.

Learning: It is conducted the same way as in Section 3.3.1. More specifically, we tune two different sets of weights, namely θ_{ρ} and θ_{Φ} , for **AM/AM** and **AM/PM DPD NNs**, respectively. To tune the weights, we use a **GD** algorithm alongside with an Adam optimizer [53]. Thus, based on equation (3.2), the updating rule becomes

$$\begin{aligned} \theta_{\rho}^k &= \theta_{\rho}^{k-1} - \eta_{\rho} \nabla_{\theta_{\rho}^{k-1}} \mathcal{L}_{\rho}(\mathcal{D}_{\rho}, \theta_{\rho}^{k-1}), \\ \theta_{\Phi}^k &= \theta_{\Phi}^{k-1} - \eta_{\Phi} \nabla_{\theta_{\Phi}^{k-1}} \mathcal{L}_{\Phi}(\mathcal{D}_{\Phi}, \theta_{\Phi}^{k-1}), \end{aligned} \quad (3.7)$$

where \mathcal{L}_{ρ} and \mathcal{L}_{Φ} are the loss functions to optimize for **AM/AM** and **AM/PM DPD NNs** respectively. For both **NNs**, we optimize a **MSE** loss function because are solving a regression problem. This learning is denoted as “conventional” learning.

Inference: It is performed according Figure 3.6. We can see that the **OFDM** signal \mathbf{i} is transformed from Cartesian to polar representation using the Cartesian-to-Polar (**C2P**) module because the **NNs** need the polar information of the signal. The **DPD AM/AM** produces the inverse **AM/AM** characteristic of the **PA** which is injected into the **DPD AM/PM**. Indeed, the **AM/PM** distortion produced by the **PA** depends on the input amplitude signal. Thus, we must consider the input amplitude provided by **DPD AM/AM** in order to have the correct phase shift compensation. The phase compensation inferred by

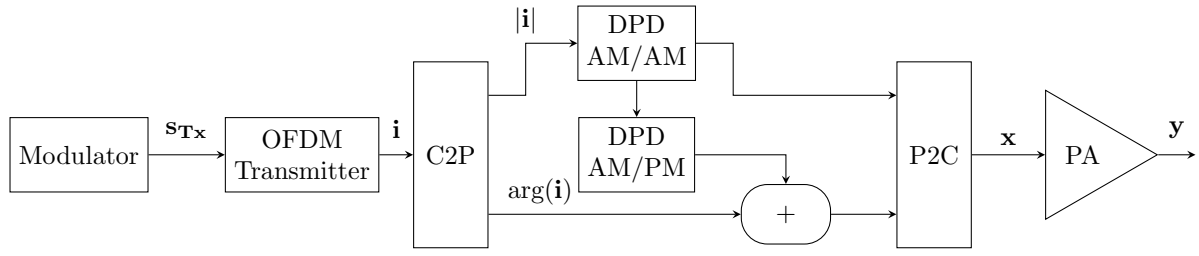


Figure 3.6: System model with DPD in real-time

the **DPD AM/PM** is then summed to the phase of the signal i . Eventually, the resulting predistorted signal is given by

$$\mathbf{x} = \widehat{f_\rho^{-1}}(|i|) \exp \left(j \left(\arg(i) + \widehat{f_{-\Phi}} \left(\widehat{f_\rho^{-1}}(|i|) \right) \right) \right) \quad (3.8)$$

where j denotes the complex number.

Similarly to the use of the **C2P** module, we use a Polar-to-Cartesian (**P2C**) module to convert the polar representation to a Cartesian representation of the signal.

Important note 3. The majority of transmitters are Cartesian. Hence, the modules **C2P** and **P2C** are likely to be used in a real-system because the **RF** chain admits only In-Phase and Quadrature (**IQ**) signal. For numerical evaluation, these modules are simply designed using the functions `real`, `imag`, `abs`, `angle` which respectively acquire the real part, imaginary part, modulus and phase of the complex signal. Eventually, it exists some polar transmitter but are less common than the Cartesian ones.

3.3.2.3 Complexity

Regarding complexity, one can evaluate the total number of Floating-Point Operations (**FLOPs**). Regarding our neural network design, the total number of **FLOPs** required to compute an inference stage is,

$$P_{LCDPDNN} = B_{DPD} \left(\underbrace{(6N_\rho + 36)}_{AM/AM} + \underbrace{(6N_\Phi + 34)}_{AM/PM} \right), \quad (3.9)$$

where B_{DPD} denotes the number of symbols during inference stage. N_ρ and N_Φ denote respectively the number of neurons, used by the **FCN** layer, in the **AM/AM** and **AM/PM** NNs. This formula takes into account the operations induced by the hidden layers, multiplications, additions and subtractions. We also take into account the operations related to activation functions. Regarding the ReLU function, it can be interpreted as 1 **FLOP** in this evaluation since it is a comparison. However, the complexity evaluation of the sigmoid is different. Indeed, it implies estimating the computational cost of the exponential function. In our simulations, we use the *TensorFlow* framework [54] to perform **NN** training and inference. Computation of exponential relies on the library *Eigen* [55] which is specialized on vector, matrix operations. In this library, exponential is evaluated by doing, $\exp(x) = 2^q \exp(r)$ where $q \in \mathbb{N}$ and $r \in \mathbb{R}$. Then $\exp(r)$ is found using a 5th-order polynomial approximation. Thus, a single sigmoid neuron leads to 27 **FLOPs**. In addition, it must be underlined that the equation (3.9) does not take into account the complexity of the Cartesian to polar conversion, for brevity.

3.3.3 Numerical simulations

We begin considering a **PA** model version of the characteristics presented in Section 2.3.2.2. Then, we assess the performance of our solution and compare it to state-of-the-art solutions. For our proposed

Table 3.3: Parameters for physical layer

Parameter	Symbol	Value
Modulation	N/A	QAM
Modulation order	M	64
Waveform	N/A	OFDM
FFT Size	N_{fft}	1024
Cyclic prefix	N_{CP}	72
Active subcarriers	N_a	666
Subcarrier spacing	f_{sc}	15kHz
Bandwidth	B_W	10MHz

NNs and the solutions proposed in [43] and [44], we consider working on the system model presented in Section 2.2. We also assume that the noises \mathbf{n} and h are negligible in order to better show the ability of correcting the PA nonlinearities. Table 3.3 presents the physical layers parameters related to the system model, which emulates a LTE system. Multiple criteria will be analyzed: EVM, spectral regrowth and ACLR. In addition, we provide all the parameters required to train and execute our proposed NNs.

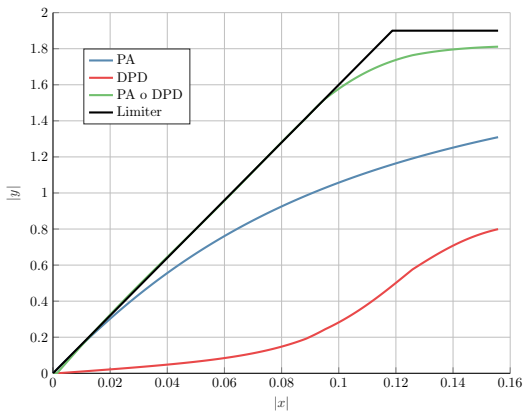
3.3.3.1 Choosing the NNs hyperparameters

Concerning the choice of NNs hyperparameters, there is no absolute rule to find the optimal hyperparameters for NNs. However, it exists some best practices to efficiently choose hyperparameters [56].

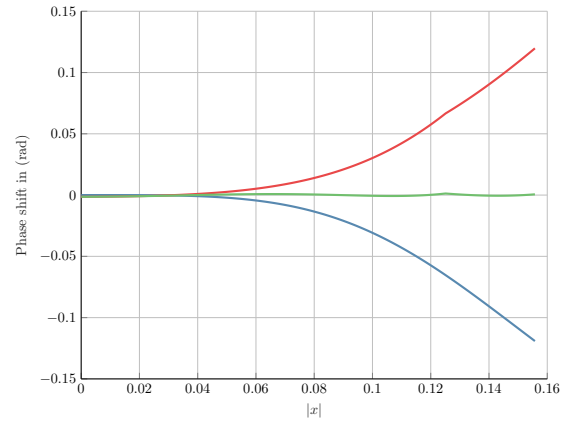
- *Total data:* To train the NNs, one often needs a large amount of data to ensure convergence. Here, we generate a signal using a uniform amplitude distribution which will push the PA into nonlinear behaviors. Thus, we choose sufficient data to obtain the optimal DPD. Since the data will be divided into batches, we must ensure that the total data is a multiple of the batch size. Here, we assume having enough data, i.e. 280576 IQ symbols which correspond to 256 OFDM symbols.
- *Learning rate:* According to equation (3.2), we can see that the learning rate η controls the convergence speed. It is one of the most important hyperparameters to tune because convergence is more sensitive to this parameter. However, there is no analytical expression to derive the best learning rate for a NN model. Here, we choose a common value for each NN, i.e. $\eta_\rho = \eta_\Phi = 10^{-3}$. Then, this learning is adapted during training using an Adam optimizer [53].
- *Batch size:* This hyperparameter allows controlling how the GD algorithm is performed. Typically, when $B_s = 1$, we perform a gradient update on each sample of the dataset. Depending on the model, it is often better to consider $B_s = N_{\mathcal{D}}$ or $B_s \ll N_{\mathcal{D}}$. Both choice allow fast convergence; the second one is better in general because it is more efficient computationally. Indeed, we only fit a small portion of the dataset in the memory and is less likely stuck in a local minimum. Hence, we consider having $B_s = 128$. Usually, the batch size is often a power of 2 to improve matrix computations on Graphics Processing Units (GPUs).
- *Number of neurons and hidden layers:* Thanks to our custom design, we only need a FCN with one hidden layer and few neurons to compute optimal DPD. The number of neurons is determined using a grid search. It consists in running several jobs of simulation changing the number of neurons and take the best one in terms of performance. We have also used this technique to

Table 3.4: Hyperparameters for DPD learning

Parameter	Symbol	Value
Total data	$N_{\mathcal{D}}$	280576 IQ symbols
Learning rate	$\eta_{\rho} \& \eta_{\Phi}$	10^{-3}
Batch size	B_s	128
Number of neurons DPD AM/AM	N_{ρ}	6
Number of neurons DPD AM/PM	N_{Φ}	4



(a) AM/AM characteristics



(b) AM/PM characteristics

Figure 3.7: AM/AM and AM/PM characteristics using with PA and DPD

choose the batch size, optimizer and learning rate. Doing so, we choose a set of hyperparameters giving optimal performance. Since, the learning stage does not require real-time considerations, one can afford to use such hyperparameters optimization.

By choosing such parameters, we ensure the convergence of the training stage with efficient resulting **DPD**.

3.3.3.2 Correction of the AM/AM and AM/PM characteristics

We choose here a value of $p = 0.7$ to model a highly nonlinear **PA** to see if our solution can correct a high degree of nonlinearity. We choose an $IBO_{dB} = 8\text{dB}$. In Figure 3.7, we present the corrections induced by the proposed **DPD**. The “Limiter” curve, in black, corresponds to a linear **PA** until its saturation characterized by $\min(G|x|, V_{sat})$ without phase distortion. Then, the blue curve represents the **PA**, the red curve presents the characteristic of the **DPD** and finally the green one is the resulting system composed of the **DPD** and the **PA**.

We can see that the use of **DPD** completely cancels the **AM/PM** distortion. There, the **DPD** has the same performance as the limiter. Concerning the **AM/AM** distortion, we can see that the resulting system **PA** and **DPD** allows converging to a linear **PA**. However, when the input amplitude exceeds 0.1, we observe a small degradation compared to the “Limiter”.

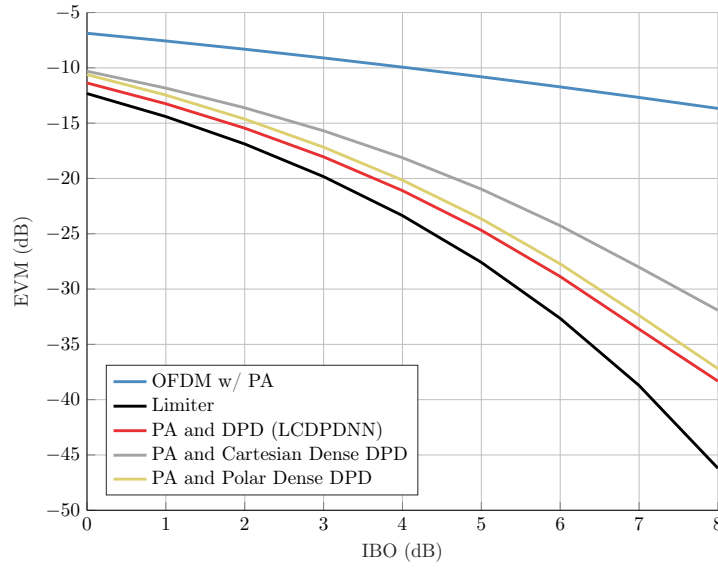


Figure 3.8: EVM performance in (dB) of LCDPDNN and literature solutions

3.3.3.3 EVM performance

To assess the performance of our solution, we evaluate the **EVM** with and without **DPD** using our proposed solution. Similarly, we use $p = 0.7$ and an **IBO** ranging from 0dB to 8dB.

Figure 3.8 presents the **EVM** versus the **IBO** using a **PA** which presents **AM/AM** and **AM/PM** distortions. The “Limiter” curve corresponds to a **PA** linear until its saturation characterized by $\min(G|x|, V_{sat})$ without phase distortion. We observe that the **PA** exhibits high nonlinearities.

Our solution, named LCDPDNN, is close to the achievable “Limiter” in terms of performance using only 10 neurons which justify the low-complexity design. Besides, we can observe a higher degradation in high **IBO** values because we learn on a highly nonlinear model. Moreover, this architecture is made to cope with nonlinear models and will be less effective on a **PA** almost linear. Furthermore, we can notice that our algorithm is better than the solutions provided in [43], referred to as “Cartesian Dense DPD”, and [44], referred to as “Polar Dense DPD”. Specifically, we have up to 6dB gain compared to the “Cartesian Dense DPD” showing that the polar approach is more efficient. The “Polar Dense DPD” is closer to our approach in terms of **EVM** performance even if a bit less efficient than our proposal. Thus, we can state that our custom design exhibits better performance than traditional solutions.

3.3.3.4 Spectral analysis

In this paragraph, we introduce spectral analysis of our DPD. Thereafter, each **PSD** figure will present a spectral analysis of the **PA** output before and after DPD usage. The “Limiter”, *i.e.* the **PA** linear until saturation, denotes the performance boundary for every numerical simulation.

Figure 3.9 presents different **PSDs** upsampled according to the left top corner scheme. The **OFDM PSD** is pictured in orange and represents our baseline. When $IBO_{dB} = 6dB$ or $IBO_{dB} = 8dB$, the **PA** induces a high level of **OOB** distortions characterized by an important spectral regrowth. In both cases, we note a significant spectral degrowth using our proposed **DPD**. We almost achieve the same performance as the “Limiter”, specially when $IBO_{dB} = 8dB$.

Then, we introduce the use of a realistic **RF** chain using Digital Up Conversion (**DUC**) and Digital Down Conversion (**DDC**) functions. These functions are composed of multiple filters which design and coefficients can be found in [57]. This allows us to test our **DPD** with realistic data converters. A simple scheme is represented in the top left corner of Figure 3.10. Similarly to the Figure 3.9, Figure 3.10 presents spectrums with $IBO_{dB} = 6dB$ and $IBO_{dB} = 8dB$. The **PA** presents the same level of **OOB**

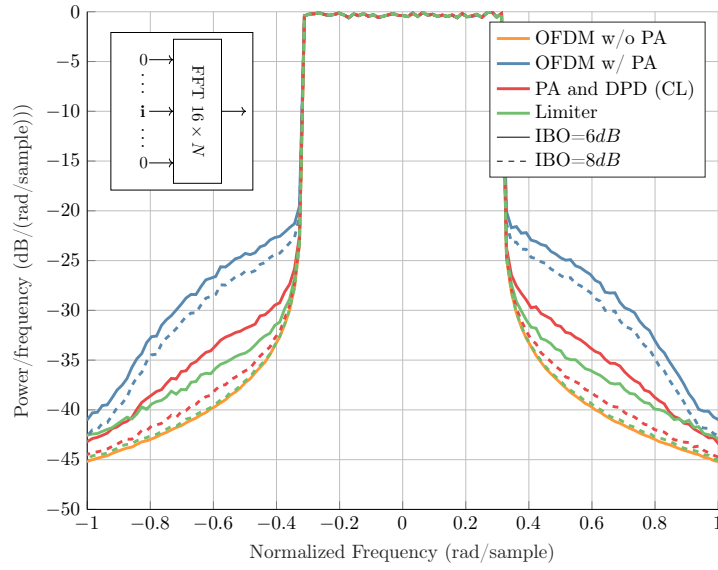
Figure 3.9: PSD before and after LCDPDNN (CL) using an oversampled FFT, $p = 0.7$

Table 3.5: ACLR in (dB)

IBO	PA	Limiter	LCDPDNN	Cartesian [43]	Polar [44]
8dB	-33	-53	-47	-42	-44

distortions as pictured on Figure 3.9. Concerning the DPD, we can state that the spectral regrowth is still considerably reduced. However, using $IBO_{dB} = 8\text{dB}$, we observe that the “Limiter” is still better than the DPD. At normalized frequency of 0.5, a spectral regrowth appears at -42dB . This slight degradation will not affect the global system performance. Thereafter, the spectral analysis will be done using the filter presented here.

Besides, in Figure 3.11, we present the performance of the solutions from [43] and [44]. We observe that these techniques both offer an important spectral degrowth compared to the OFDM with PA only. However, our proposed solution still performs better to linearize PA nonlinearities *w.r.t.* to cited works.

Furthermore, one can evaluate the ACLR based on the presented PSDs in Figure 3.11 and equation (2.9). The left ACLR using each solution is given in Table 3.5. Evaluating the right ACLR is useless in that case since the PSD is symmetric. First, our solution reaches an ACLR of -47dB which is sufficient to comply with the LTE standard. Yet, there is still a gap compared to the “Limiter”. State-of-the-art solutions also reach a low ACLR, but our proposal remains the best as well regarding this figure of merit.

3.3.3.5 Discussion

Here, we present a synthesis of our DPD and the ones from the literature in terms of figures of merit and complexity. Table 3.6 summarizes all these key values to assess which DPD is the best. In this table, we added the proposals from [46] and [45] to compare their complexity and relative gain to our proposition. Besides, the ACLR gain is the difference between the ACLR using the PA and the one considering the DPD. Based on this table, we can clearly see that our solution has the best performance in every performance indicator. Concerning the “Polar Dense DPD” from [44], increasing the number of neurons can lead to the same performance as our proposal, but this increases its complexity. Besides, the NFLOPs column shows the normalized number of FLOPs regarding the solution proposed in [46].

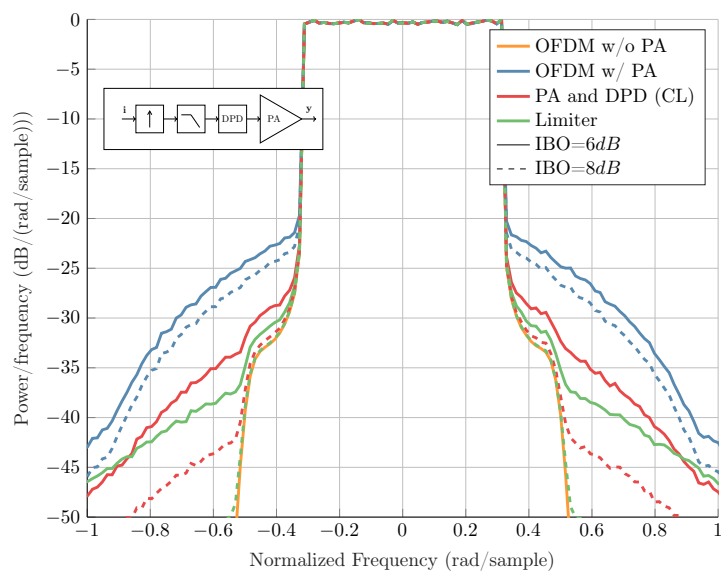


Figure 3.10: PSD before and after LCDPDNN (CL) with a digital up converter, $p = 0.7$

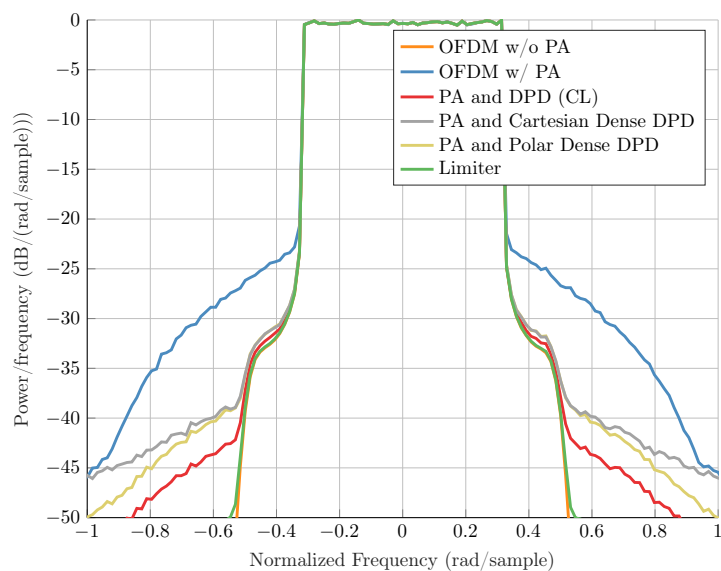
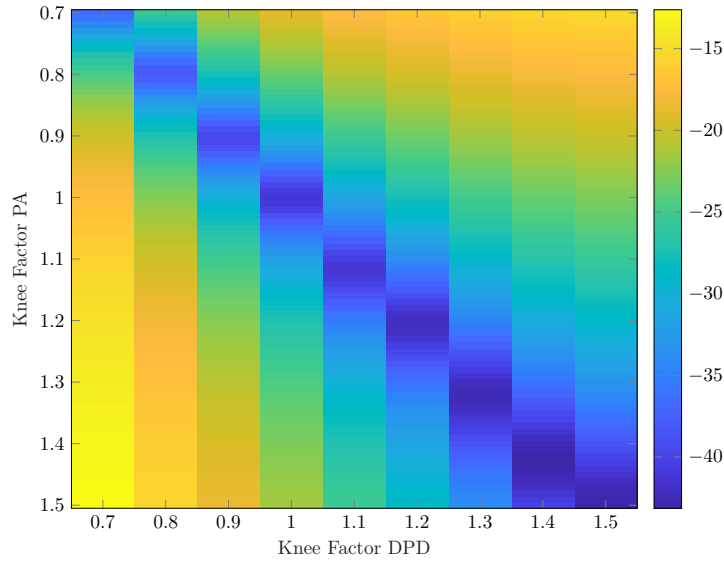


Figure 3.11: PSD using presented DPD and state-of-the-art solutions with $IBO_{dB} = 8dB$, $p = 0.7$

Table 3.6: Comparison of DPD algorithms

Related Work	ACLR Gain	EVM	FLOPs	NFLOPs
RT2DNN [46]	12dB	N/S	412	1
DPD w/o ILA [45]	N/S	N/S	184	0.45
Cartesian Dense DPD [43]	9dB	-32dB	322	0.78
Polar Dense DPD [44]	11dB	-37dB	162	0.39
LCDPDNN	14dB	-38dB	142	0.34

Figure 3.12: EVM performance in (dB) with $0.7 \leq p \leq 1.5$, $IBO = 8\text{dB}$

Thus, we achieved to propose an efficient custom design **DPD** based on **NNs** which exhibits low-complexity. Besides, such improvement in terms of complexity allows to consider this solution as **DPD** candidate for **6G**.

3.3.3.6 Limitations

In the previous paragraphs, we saw that our **LCDPDNN** provides excellent performance while respecting a low-complexity design. Nonetheless, one objective of this thesis work is to propose a **DPD** robust to time-varying effects affecting the **PA**. Therein, we evaluate the ability of our **NNs** to adapt to time-varying effects. Thus, we consider an adaptive scenario where the parameter p of the **PA** model is varying – see equation (2.3).

Figure 3.12 presents the **EVM** performance considering that p is varying with $IBO_{dB} = 8\text{dB}$. The x axis corresponds to a range of **DPD** trained for $p = 0.7 + 0.1k, k \in [0, 8]$ and the y axis corresponds to the value of p during performance test with $p = 0.7 + 0.01k, k \in [0, 80]$. Thus, during this simulation, we evaluate for each **DPD** function the **EVM** performance considering that the **PA** is varying, *i.e.* p is varying.

We can observe that the **NN** achieves its optimal on the diagonal which correspond to a trained p for the same inferred p . Moreover, we can underline the fact that the performance is severely degraded when a trained value of p is inferred on another value. Hence, the model is not able to adapt to a model variation. However, it could be envisaged to retrain the model online, but it would consume a lot of

symbols for online training phase. Indeed, we also seek small data usage in order to minimize global latency of the system during online calibration. Therefore, it is essential to find other solutions to improve the adaptivity of the DPD.

3.4 Conclusion

In this chapter, we proposed custom NNs architectures to perform DPD to linearize PAs. In particular, we split the correction of the PA distortions, *i.e.* we have dedicated NNs correcting the AM/AM and AM/PM distortions, respectively. In doing so, we drastically improve the performance of the DPD in terms of EVM and ACLR. Then, this gives us an efficient DPD to linearize the PA. This approach has already been investigated in [44]. However, due to the “blackbox” design of the solution, the resulting complexity is high and likely not suited to real-time applications. In our custom design, we optimized every operation to lower the complexity of the resulting DPD functions. Thus, we have shown that using a custom design rather than a “blackbox” design offers a better tradeoff between performance and complexity. Specifically, we have a better performance at a lower complexity.

Despite the improvements brought by the LCDPDNN, our solution still severely lacks of adaptivity as evaluated in the numerical simulations of this chapter. Indeed, the characteristics of a PA are likely to change over time because of external effects such as the temperature. The temperature can modify directly the distortions induced by the PA. The “conventional” training used in this chapter is considered as static and consumes a lot of data to be retrained. This implies, first the incapacity to adapt to changes in the characteristics of the PA leading to a poor performance of the DPD. Second, a high-latency due to the data-consuming algorithm which makes the DPD unsuitable for wireless communications systems where latency must be low. Therefore, some issues are still to be resolved: *can we improve our solution to make it resilient to time-varying effects affecting the PA?*, and *can we improve the adaptivity of our solution using few data?*.

Thus, in the next chapter, we investigate solutions to address the aforementioned issues. This implies, finding algorithms performing low-cost online training and compatible with our custom NNs designs.

The technical content of this chapter has been disseminated by the following conference papers and journal paper in part:

- [C1] E. Calvanese Strinati, D. Belot, **A. Falempin** and J-B. Doré, “Toward 6G: From New Hardware Design to Wireless Semantic and Goal-Oriented Communication Paradigms,” In Proc. IEEE European Solid State Circuits Conference (ESSCIRC), Grenoble, FR, pages 275-282., 2021.
- [C2] **A. Falempin**, J-B. Doré, R. Zayani and E. Calvanese Strinati, “Low-Complexity Adaptive Digital Pre-Distortion with Meta-Learning based Neural Networks,” In Proc. IEEE Consumer Communications & Networking Conference (CCNC), NV, USA, pages 453–457., Feb 2022.
- [J1] **A. Falempin**, J-B. Doré, R. Zayani and E. Calvanese Strinati, “Low-Complexity Adaptive DPD: From online optimization to meta-learning,” IEEE Transactions on Broadcasting, 2022.

Learning how to Learn the Digital Pre-Distortion

Contents

4.1	Introduction	42
4.1.1	Motivations	42
4.1.2	Related Work	42
4.1.3	Contributions	43
4.2	Meta-Learning : A new way of optimizing neural networks	43
4.2.1	Model-Agnostic Meta-Learning (MAML) for DPD	43
4.2.2	Dataset construction for DPD	47
4.2.3	Numerical simulations	48
4.2.4	Complexity	50
4.3	Efficient Weights Selection for Adaptive DPD	51
4.3.1	DPD Neural Network Weights Selector (DPDNNWS)	51
4.3.2	Numerical simulations	52
4.3.3	Complexity	55
4.4	Comparison between DPD algorithms	55
4.4.1	Spectral analysis and ACLR	55
4.4.2	Synthesis on all performance metrics	56
4.5	Conclusion	56

4.1 Introduction

THIS chapter tackles the issue of adaptivity needed in our proposed Digital Pre-Distortion (DPD). Indeed, PAs have time-varying AM/AM and AM/PM characteristics, due to multiple factors such as the power variation and temperature. Consequently, an adaptive DPD is needed in order to tackle this issue. In the previous chapter, we proposed a DPD based on Neural Networks (NNs) able to linearize the PA nonlinearities while keeping a low-complexity design. However, this solution suffers from a lack of adaptivity, making it unable to deal with time-varying aspects. In this chapter, we explore a new class of training algorithms for NNs, named meta-learning which allows improving NN generalization. Then, we investigate a NN based DPD using meta-learning offering satisfying performance *w.r.t.* to any PA characteristic. In addition, we also provide an algorithm realizing a selection of the best parameters for the NN. This approach is based on Generalized Likelihood Ratio Test (GLRT) and a priori known PA characteristics with their associated NN DPD weights.

4.1.1 Motivations

A PA exhibits multiple nonlinear effects, AM/AM and AM/PM distortion as well as memory effects that are not covered in this thesis. In the literature, we easily find approaches which propose solutions to linearize the distortions of the PA taking into account the memory effects. However, a main concern is often neglected: the time-varying aspect of the PA. Indeed, the characteristic of the PA may vary because of external effects, mainly temperature, output impedance and power variations. Therefore, most state-of-the-art solutions will fail to linearize the PA efficiently at a low cost. Still, one solution could be re-training the DPD from scratch, but this needs lot of data and computation time.

Thus, our goal is to propose a DPD that can adapt quickly to a change of state of the PA. We also want to minimize the data necessary to perform the calibration of our DPD. Then, the problem to solve is twofold: i) the DPD shall be able to adapt to a new state of the PA, ii) this adaptation shall consume few data to ensure a low-latency adaptation.

4.1.2 Related Work

As stated in the previous chapter, there are plenty of works which propose solutions to perform DPD for PA linearization. However, to that extent, very few works dig into the adaptive aspect of the DPD. The classical approach to perform an adaptive DPD is to use Indirect Learning Architecture (ILA) and trigger a new training periodically. Most solutions use Look-up Tables (LUTs) or memory-polynomial models to model their DPD [58], [59]. These solutions often exhibit high computational cost due to the use of memory-polynomial model and large size LUTs. Moreover, they do not provide an efficient approach to cope with time-varying aspects.

Conversely, an adaptive DPD based on Direct Learning Architecture (DLA) and ILA has been proposed in [60]. This solution uses the cascade of two DPDs and a recursive least squares algorithm to enable adaptive DPD. The results are promising but the envisaged scenario presents few variations between each PA characteristic and the algorithm complexity may still be challenging due to the memory-polynomial design of the DPD. In [43] and [44], authors proposed deep learning Fully Connected Network (FCN) designs using ILA to perform DPD. These solutions do provide an adaptive behavior but need a large amount of data to converge and may suffer from slow convergence, and therefore high latency issues. Other literature solutions using NN do not provide any adaptive behavior to the DPD.

Besides, considering NNs, meta-learning, a new class of training algorithms, has recently emerged to improve generalization of NNs. This kind of algorithms is widely used in computer vision and robotic fields. Nevertheless, some works used meta-learning in signal processing. Specifically, the work done in [61] demonstrates that meta-learning greatly reduce training overhead and complexity. Furthermore, works in [62] and [63] have investigated the use of meta-learning algorithms to perform end-to-end learning with few pilots demodulation. Hence, it is proven that meta-learning can significantly reduce

data consumption and training time. Nevertheless, for instance, meta-learning is not really used in wireless communications, and no literature has been found to perform adaptive DPD using meta-learning approach.

4.1.3 Contributions

The contributions of this chapter are twofold:

- **Adaptive DPD using meta-learning:** we provide an adaptive behavior to the DPD developed in Chapter 3. To do so, we use a meta-learning algorithm called Model-Agnostic Meta Learning (MAML) [64]. This algorithm allows to find the best initialization NN weights to converge quickly during inference stage, and particularly with unseen states during learning. Using this approach, we enable adaptivity for our DPD using few data, resulting in low-cost adaptation.
- **Weights selection for adaptive DPD:** this approach is based on the selection of appropriate NN DPD weights using hypotheses tests between the current PA state and a priori known PA characteristics. This algorithm relies on the use of GLRT. This method remains simpler than MAML in terms of hardware implementation and complexity.

The technical content of this chapter is based on [65] and [48]. The remainder of this chapter is organized as follows. Section 4.2 presents the meta-learning algorithm we use, to perform adaptive DPD, as well as related numerical simulations. Then, Section 4.3 gives another approach, referred to as weights selector, to adapt the DPD regarding a change of the PA state. We also provide numerical simulation analysis related to the use of this algorithm. Eventually, Section 4.4 gives a comparison of each DPD technique in terms of performance, complexity and adaptivity. Finally, some concluding elements are raised in Section 4.5.

4.2 Meta-Learning : A new way of optimizing neural networks

Meta-learning is an approach putting forward the “learning how to learn” concept, in other words improving a learning algorithm through multiple trainings [66]. Conversely, “conventional” learning, proposed in Chapter 3, improves NN predictions using only a single training on multiple batches of data. Thus, “conventional” learning is said to be task specific, *i.e.* it will learn from one configuration only. Contrary to the latter, meta-learning learns on multiple configurations called tasks. In this work, a task will be referred to as a characteristic of the PA at a specific time. By doing meta-learning, one can quickly train and adapt a NN model *w.r.t.* to a new task from few data.

Thus, we present here a meta-learning approach to improve the adaptivity of NN based DPD. We first review a recent meta-learning algorithm called MAML to improve NN generalization. Second, we conduct numerical evaluation to assess the performance of the algorithm using few data and few steps during training.

4.2.1 Model-Agnostic Meta-Learning (MAML) for DPD

MAML is a meta-learning algorithm that enables fast adaptation of neural networks. Its specificity is that it can be used on whatever NN model that is trained using gradient descent. So, it is suitable to use this method for our custom DPD to avoid architecture constraints. The objective of MAML consists in finding the best weights initialization for a NN to perform fast inference based on unseen data.

In “conventional” learning, we are trying to minimize a loss function *w.r.t.* to NN weights and provided dataset. In other words, we are solving the following problem:

$$\arg \min_{\theta} \mathcal{L}(\theta, \mathcal{D}) \quad (4.1)$$

As stated in Chapter 3, we use **GD** algorithm to solve this optimization problem. However, the weights θ are randomly initialized. This leads to slow convergence during training and may also conduct to poor generalization. Thus, **MAML** acts as a weights' initializer, *i.e.* we want to find θ^0 , –see equation (3.2).

For better understanding of meta-learning and especially **MAML**, we introduce the notion of task and how it is used in meta-learning. Then, we explain how to find the weights' initialization. Finally, we dig into the adaptation process from.

4.2.1.1 Tasks

Formally, a task \mathcal{T} is characterized by a dataset, that is divided according to algorithm needs, *e.g.* training and validation sub-datasets. We also encapsulate the loss function $\mathcal{L}(\mathcal{D}, \theta)$ associated to the dataset. As an example, “conventional” learning will use a single task. For our **NN DPD**, let's consider only the **NN** performing the **AM/AM** distortion correction. We then define $\mathcal{T} = \{\mathcal{D}, \mathcal{L}\}$ where $\mathcal{D} = (|\mathbf{z}|, |\mathbf{x}|)$ and \mathcal{L} is a **MSE** loss function given by

$$\mathcal{L}(\mathcal{D}, \theta) = \frac{1}{B_s} \sum_{j=1}^{B_s} \left(|\mathbf{x}_j| - \widehat{f}_\rho^{-1}(|\mathbf{z}_j|) \right)^2. \quad (4.2)$$

Here, θ correspond to θ_ρ , defined in Chapter 3, for better reading. It is implicitly included in \widehat{f}_ρ^{-1} , which the prediction of the **NN** for **AM/AM DPD** using the weights θ .

Thus, a task is just a way to represent a dataset passing through a **NN** with specific weights and loss. Now, if we expand this concept to meta-learning, it is roughly the same thing except that we consider a distribution of tasks, referred to as $p(\mathcal{T})$. A distribution of tasks is characterized by the variation of one or multiple parameters within a specific range, affecting the dataset of values fed to the **NN**. In other words, for each sampled $\mathcal{T}_i \sim p(\mathcal{T})$, we have a specific dataset \mathcal{D}_i and loss \mathcal{L}_i , *i.e.* $\mathcal{T}_i = \{\mathcal{D}_i, \mathcal{L}_i\}$.

Thereafter, we consider a distribution of tasks $p(\mathcal{T})$ characterized by the variation of $p \in [0.7, 1.5]$ –see equation (2.3). Figure 4.1 presents a graphical representation of tasks sampled from $p(\mathcal{T})$. We can see that, the chosen distribution models the nonlinear behavior of the **PA**, *i.e.* more or less linear. Regarding the loss function, equation (4.2) becomes

$$\mathcal{L}_i(\mathcal{D}_i, \theta_i) = \frac{1}{B_s} \sum_{j=1}^{B_s} \left(|\mathbf{x}_j| - \widehat{f}_{\rho,i}^{-1}(|\mathbf{z}_j|) \right)^2. \quad (4.3)$$

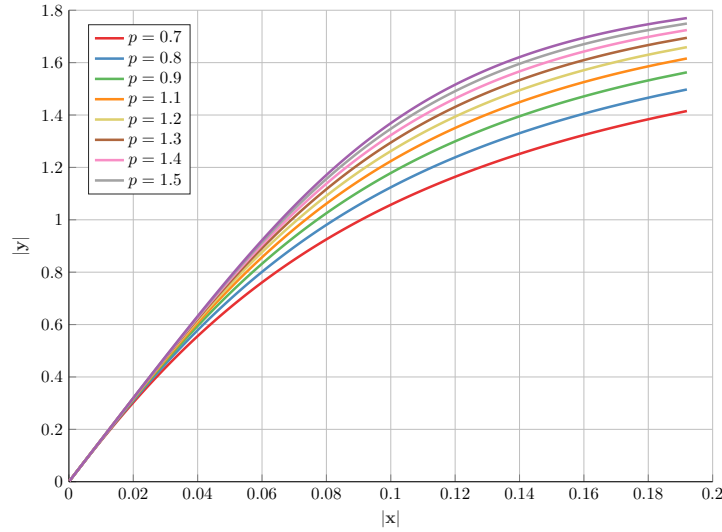
The loss function remains the same as before, except that it is parameterized by the task \mathcal{T}_i .

4.2.1.2 Learning how to learn

Performing a meta-learning requires two major steps, i) defining on which task distribution we learn, ii) what are we learning and how. Here, let us explain these steps regarding the **MAML** algorithm.

Tasks: In the previous paragraph 4.2.1.1, we gave insights on tasks and related distribution. One must choose carefully the task distribution because meta-learning algorithms and especially **MAML** are very sensitive to the data. Choosing the task distribution mainly depends on the problem to solve. The issue to tackle must be parameterized by one or multiple variables. For meta-learning, training is conducted over the tasks sampled from $p(\mathcal{T})$. Moreover, the dataset \mathcal{D}_i is divided into two sub-datasets, \mathcal{D}_i^{tr} and \mathcal{D}_i^{te} . It permits increasing generalization during the training phase.

As a toy example, suppose we want to derive the function $y = \omega_A \sin(x + \omega_\phi)$ where ω_A and ω_ϕ are respectively the amplitude and the phase given in the range Ω_A and Ω_ϕ . Then, the task distribution shall take into account both varying aspects of the phase and amplitude, and is then characterized by

Figure 4.1: Representation of PA tasks with $p \in [0.7, 1.5]$

the ranges Ω_A and Ω_ϕ . Then, this implies that each task is represented by a dataset having random values of x and y , respecting the aforementioned ranges.

Now, concerning our case, to perform our **DPD**, we consider thereafter working only with the **AM/AM NN DPD**. In practice, the time-varying effects usually affect the gain of the **PA**, *i.e.* the **AM/AM** characteristic. Then, it is more valuable to learn how to cope with the **AM/AM** time-varying aspect.

Important note 4. *Defining the tasks is the most crucial step, therefore one must be careful regarding the chosen distribution and size of the datasets belonging to each task.*

Learning to learn: In meta-learning, we often learn a learning algorithm instead of learning the weights that answer to a single-task problem. It exists different meta-learning algorithms, yet we choose **MAML** because it does not require any specific architecture and can be used straightforward with any **NN** trained using a **GD** algorithm. The objective of **MAML** is finding the best weights to initialize a **NN** regarding a task distribution, such that the **NN** can adapt to an unseen task using few data and little training time. We call this step meta-training.

Remark 6. *Meta-learning is often compared to Mutli-Task Learning (MTL). However, MTL just tries solving multiple problems at once, but it does not consider solving an unseen task sampled from a distribution of tasks. It will then be less efficient for a time-varying system.*

Hereafter, we denote by θ , the weights inferred by **MAML** algorithm. **MAML** works by finding optimal weights ϕ_i across tasks sampled from $p(\mathcal{T}_i)$ using **GD**. Then, those weights are used to derive θ , our optimized initialization weights for quick convergence inference. Figure 4.2 presents the **MAML** algorithm composed of three main steps:

- Step 1:* Sampling a batch of tasks \mathcal{T}_i from $p(\mathcal{T})$
- Step 2:* Find task-specific weights ϕ_i using **GD** algorithm with \mathcal{D}_i^{tr} . It is represented as the “Inner Loop” on Figure 4.2.
- Step 3:* Find weights θ by minimizing the sum of losses across all tasks \mathcal{T}_i w.r.t. to \mathcal{D}_i^{te} . It is represented as the “Outer Loop” on Figure 4.2.

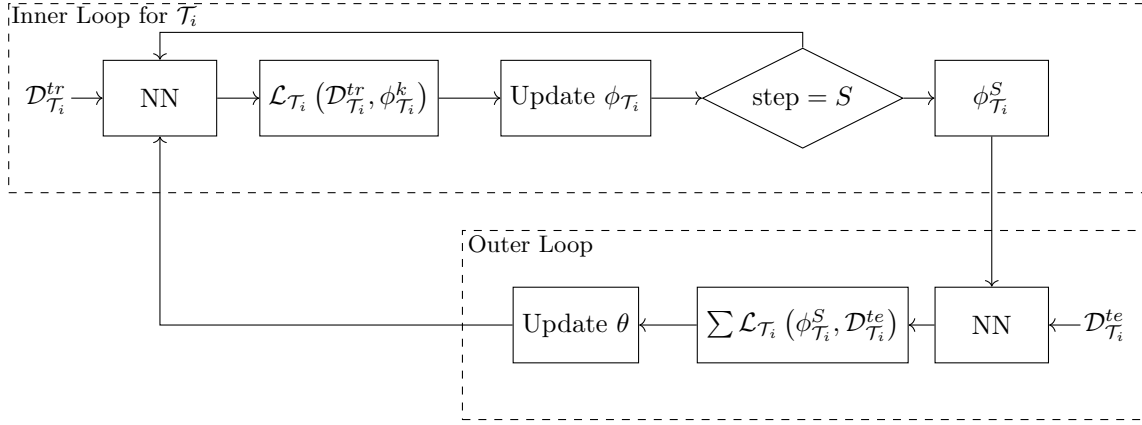


Figure 4.2: MAML algorithm state machine

Concerning *step 2*, we solve the same minimization problem as in (4.1) for \mathcal{T}_i , which gives

$$\phi_i = \arg \min_{\theta} \mathcal{L}_i(\mathcal{D}_i^{tr}, \theta). \quad (4.4)$$

To solve this optimization problem and derive ϕ_i , we perform a **GD**. Considering one gradient step,

$$\phi_i = \theta - \nu \nabla_{\theta} \mathcal{L}_i(\mathcal{D}_i^{tr}, \theta) \quad (4.5)$$

where α represents the “inner” learning rate. This update process is often called meta-update or “inner” loop as pictured in Figure 4.2. As a comparison, it can be seen as “conventional” on each task \mathcal{T}_i .

Next, from all the task-specific weights ϕ_i , **MAML** realizes a second optimization to find the weights θ . Then, the problem to solve becomes

$$\arg \min_{\theta} \sum_{i=1}^{N_{\mathcal{T}_i}} \mathcal{L}_i(\mathcal{D}_i^{te}, \phi_i^S) \quad (4.6)$$

where $N_{\mathcal{T}_i}$ is the number of tasks.

Remark 7. In problem (4.6), instead of a single-task optimization as in (4.4), we minimize the sum of losses w.r.t. each task \mathcal{T}_i using task-specific weights ϕ_i derived in the “inner” loop. Finally, the loss is evaluated w.r.t. \mathcal{D}_i^{te} to improve generalization of the algorithm.

Solving this problem still uses **GD** algorithm except that we sum the loss gradients across all tasks \mathcal{T}_i . Then θ is given by,

$$\theta = \theta - \eta \nabla_{\theta} \sum_{i=1}^{N_{\mathcal{T}_i}} \mathcal{L}_i(\mathcal{D}_i^{te}, \phi_i^S) \quad (4.7)$$

By summing all the losses across tasks, we get information from every task. It means that the “outer” level, called meta-update, is trying to find the weights that are “close” to each task and more widely to the task distribution. Thus, **MAML** solves a bi-level optimization problem which allows finding the best initialization weights for an unseen task sampled from $p(\mathcal{T})$. Doing so, we minimize the time and amount of data needed to converge to optimal weights during adaptation stage.

Remark 8. Equation (4.7) involves computing second-order derivatives. Indeed, the evaluation of the loss in the “outer” level uses the task-specific weights ϕ_i derived from θ , obtained in the “inner” level. Thus, **MAML** algorithm exhibits a high complexity regarding the weights update and may

consume more data to derive θ .

Remark 9. Since our designed **NN** for **DPD** have few parameters, the second-order gradients computations will not affect the convergence that much. However, for wider **NN** architectures, it is possible to make a first-order approximation of the gradients to lower the complexity of the algorithm.

Eventually, in a practical system, we would make several acquisitions of the **PA** characteristics under different conditions, e.g. temperature variation. Then, we apply the **MAML** algorithm on a computing machine to derive the initialization weights θ . Those weights will then serve to perform the online adaptation.

Important note 5. We presented the **MAML** algorithm in its generic form. To apply it to our **DPD**, we consider the task distribution $p(\mathcal{T}_p)$ previously defined. Then, the task dataset become $\mathcal{D}_{\rho,i}$ and the chosen loss is $\mathcal{L}_{\rho,i}$.

4.2.1.3 Online adaptation

Once **MAML** has produced the initialization weights, we can use as an initializer for our **NN** instead of using default ones. By doing so, we can easily adapt to a new unseen task \mathcal{T} , but still belonging to the same task distribution as the one used during the bi-level optimization, i.e. $\mathcal{T}_n \sim p(\mathcal{T})$. This step is straightforward as it represents a “conventional” learning. The only difference comes from the weights initialization that is provided by **MAML**. Thus, the adaptation rule is given by,

$$\phi^k = \phi^{k-1} - \xi \nabla_{\phi^{k-1}} \mathcal{L}(\mathcal{D}^{tr}, \phi^{k-1}), \quad (4.8)$$

with $k = 1, \dots, S_n$ the number of gradient steps, $\phi^0 = \theta$ and \mathcal{D}^{tr} represent the required samples to perform the online training and update the model to adapted weights. ξ is the learning rate used for the **GD** algorithm.

4.2.2 Dataset construction for DPD

As stated before, the choice of tasks is important to perform **MAML**. Thus, one must construct a correct distribution of **IQ** symbols which will efficiently stimulates the **PA**. In our simulations, the best data distribution to conduct learning is the uniform distribution because it excites the **PA** in every zone with the same probability.

Thus, during online usage, from the **OFDM** symbols, we must reconstruct a uniform distribution. This will help **MAML** to converge better and then provide better **DPD** in terms of performance. To do so, we consider that our **PA** operates at a certain IBO_{dB} . This gives us an input amplitude interval for our **OFDM** symbols. We denote respectively the minimum and maximum of this interval by a and b . We subdivide this interval in Q intervals leading to a step $I_s = \frac{b-a}{Q}$. Then, each interval can be expressed as $[a + I_s k, a + I_s(k + 1)]$, $k < Q$. In order to reconstruct the chosen discrete distribution, we must have the same number of samples in each interval. In addition, in each k interval, we shall have $\frac{N_{\mathcal{D}^{tr}}}{Q}$.

We present an algorithm that will reconstruct a discrete uniform distribution based on input/output symbols of the **PA** and $N_{\mathcal{D}^{tr}}$. At initial stage, we have 8 empty bins to fill, i.e. intervals. Upon a new input sample, we take its amplitude and check if the corresponding bin is full; if not, we add the input and corresponding samples to the dataset \mathcal{D}^{tr} . The algorithm finishes once \mathcal{D}^{tr} has reached the desired number of symbols.

Table 4.1: Meta-learning parameters

Parameter	Symbol	Value
Tasks number	$N_{\mathcal{T}_i}$	17
Inner gradient steps	S	3
Batch Size	B_s	1000
Total data	N/S	51000
Epochs	N/S	1000
Optimizer	N/S	Adam
Inner learning rate	ν	Adaptive
Outer learning rate	η	Adaptive

4.2.3 Numerical simulations

Similarly to Chapter 3, we consider a PA model which characteristics are presented in Section 2.3.2.2. We give the meta-learning parameters in order to execute the MAML algorithm. Then, we assess the performance of MAML in terms of EVM and spectral analysis. The physical layer parameters are the same as the one used in Chapter 3 – see Table 3.3.

4.2.3.1 Meta-learning parameters

Table 4.1 presents all the parameters required in order to perform the MAML algorithm. The architectural design of the NNs remains unchanged. Therefore, readers may refer to Table 3.4 regarding the number of neurons and activation functions.

Choosing meta-learning parameters can be hard since the algorithm convergence highly depends on each one of them. Specifically, one shall pay attention to the task distribution, number of tasks and the number of gradient steps in the “inner” loop. Concerning the task distribution, we use $p(\mathcal{T})$ previously defined in Section 4.2.1.1, where $\rho \in [0.7, 1.5]$. This range ensures having sufficient nonlinearities to perform DPD. Then, the number of tasks required to ensure convergence of the MAML is evaluated empirically under some constraints. Indeed, we want to minimize the required amount of data for training and inference of the DPD. Hence, the number of tasks shall be low. Besides, the “inner” gradient steps are also important because it directly impacts the convergence time of the algorithm. Ideally, S should be small to avoid problems such as vanishing gradients or instability. Thus, to find both $N_{\mathcal{T}_i}$ and S , we realize a grid search among these parameters described in Section 3.3.3.1. Eventually, to improve convergence and stability of the MAML algorithm, the condition $\eta < \nu$ must be satisfied. The “outer” loop produces the new initialization weights for each task. Having a lower learning rate here will slow down convergence, but it avoids skipping the minimum of the loss function.

Besides, building the database is also challenging. For each task, we have to build the datasets \mathcal{D}_i composed of the sub-datasets \mathcal{D}_i^{tr} and \mathcal{D}_i^{te} . Then, $\mathcal{D}_i^{tr} = (|\mathbf{y}_{tr}|, |\mathbf{x}_{tr}|)$ and $\mathcal{D}_i^{te} = (|\mathbf{y}_{te}|, |\mathbf{x}_{te}|)$ with $\mathbf{x}_{tr}, \mathbf{y}_{tr} \in \mathbb{C}^{1 \times B_{tr}}$ and $\mathbf{x}_{te}, \mathbf{y}_{te} \in \mathbb{C}^{1 \times B_{te}}$. B_{tr} and B_{te} denote the datasets sizes, and we consider having $N_{\mathcal{D}_{tr}} = N_{\mathcal{D}_{te}}$. This value especially controls the generalization of the model. As the other meta-learning parameters, we use a grid search to find the right number of IQ symbols $N_{\mathcal{D}_{tr}}$ with $N_{\mathcal{D}_{tr}} = B_s \times S$, $S \in [1, 10]$ where S is the number of gradient steps, and B_s the batch size that is used in the “inner” loop. It appears that using $S = 3$ gives the best convergence and performance. Besides, if $N_{\mathcal{D}_{tr}} < 1000$, the training may never converge due to insufficient data to represent the PA, eventually leading to a poor DPD.

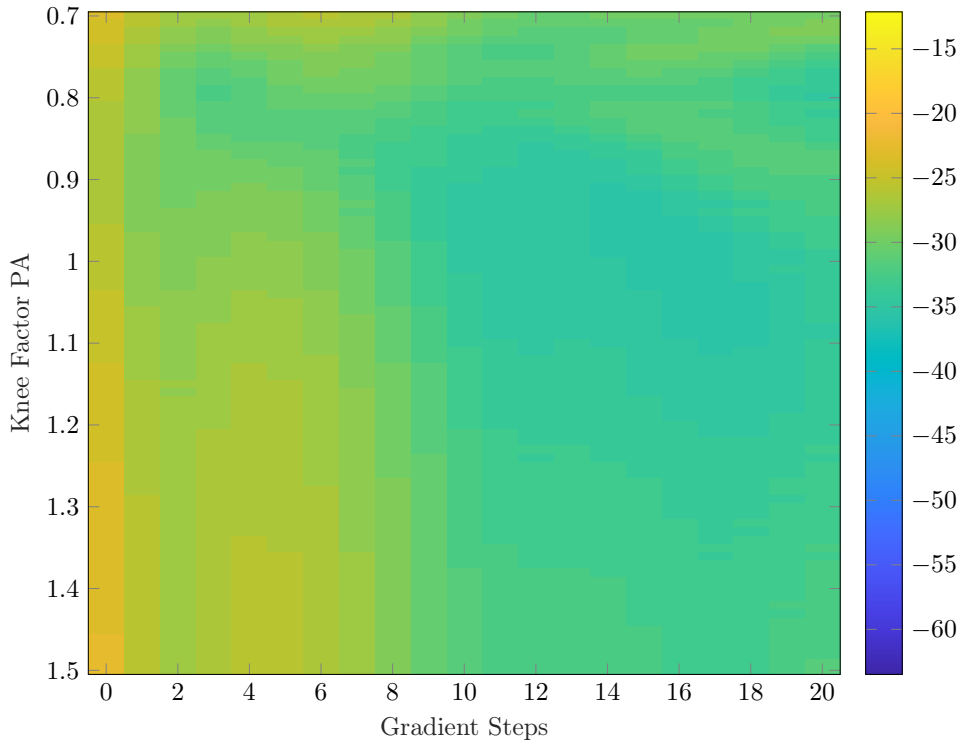


Figure 4.3: EVM performance in (dB) with $0.7 \leq p \leq 1.5$, $IBO_{dB} = 8\text{dB}$

Finally, to perform the adaptation, a dataset of 3000 IQ symbols is enough. This dataset will then be divided into batches according to the number of gradient steps.

4.2.3.2 EVM performance

To assess the performance of the meta-learning approach, we evaluate the EVM w.r.t. the chosen task distribution. Thus, we consider the variation of the “knee factor” of our PA based on Rapp model – see equation (2.3).

Figure 4.3 presents the EVM performance of MAML based DPD over a range of PAs with $p = 0.7 + 0.01k, k \in [0, 80]$ in function of the gradient steps. First, we observe that without any retraining, i.e. gradients steps equals 0, MAML reaches an average of -24dB of EVM. This performance is fair considering that the “conventional” learning can only reach -10dB in some cases –see Figure 3.12.

Besides, increasing the gradient steps notably improve the performance in terms of EVM. Indeed, using 13 gradient steps, the EVM is now ranging from -29dB to -35dB regarding the value of p . This represents a descent performance regarding the wide range of treated PAs. Actually, the performance satisfies the EVM requirements for LTE and 5G technologies, –see Table 2.1. It shall be reminded that MAML only uses 17 tasks for training. Conversely, the inference stage is conducted over 80 PAs models. Then, according to Figure 4.3, MAML can adapt to unseen tasks, but still belonging to the same task distribution. Thus, it can be said that the latter has successfully enabled an adaptive behavior for our DPD.

4.2.3.3 Spectral analysis

We report here the spectral analysis of our DPD using MAML. Similarly to the previous analysis conducted in the previous chapter, we consider using the same DUC and DDC functions. The latter are composed of filters whose characteristics can be found in [57]. The scheme of the used DUC function is pictured on Figure 3.10.

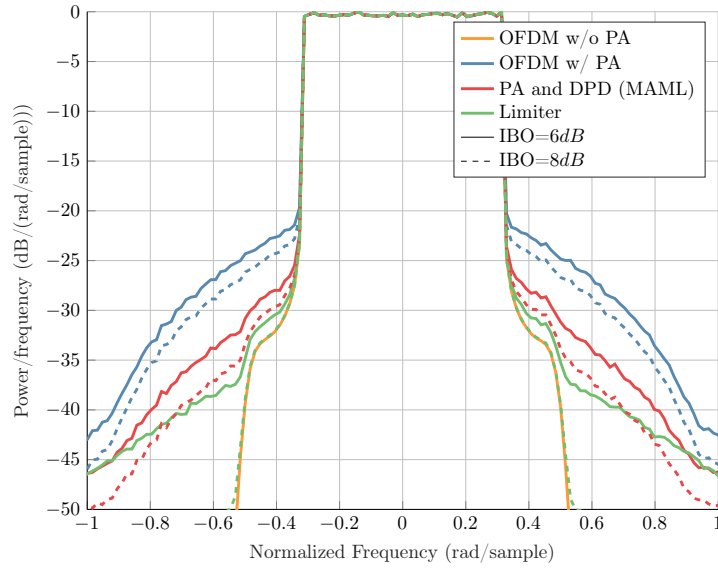


Figure 4.4: PSD before and after LCDPDNN (MAML) $p = 0.7$, $IBO_{dB} = 8dB$

Figure 4.4 presents different PSDs using $IBO_{dB} = 6dB$ and $IBO_{dB} = 8dB$. The OFDM is pictured in orange and represents the baseline. We then have the “Limiter” representing the best achievable performance. Then, we have the considered PA with $p = 0.7$, operated at $IBO_{dB} = 6dB$ and $IBO_{dB} = 8dB$. Finally, the performance of the LCDPDNN using MAML is represented in red.

We observe that using MAML, we achieve fair correction of the spectral regrowth. The majority of OOB distortions exhibited by the PA are greatly reduced by the MAML based DPD. This result is satisfying considering that the use of MAML produces a generalized model converging in few gradient steps. Thus, we can say that the use of meta-learning is relevant to perform fast DPD adaptation.

4.2.4 Complexity

The algorithmic complexity of MAML shall be also discussed, since it is perceived as a real-time algorithm for online DPD. The complexity will be evaluated in terms of FLOPs. First, MAML is based on the use of the LCDPDNN. Then, inference stage will exhibit the same complexity, i.e. 142FLOPs/symbol – see details in (3.9). Second, regarding the usage of MAML to infer initialization weights, its complexity is high, but it can be executed offline. Indeed, this stage is not meant for online usage but will facilitate the online calibration and reduce its complexity.

In order to evaluate the complexity of the online usage, one must evaluate the complexity of the “conventional” learning. Thus, one must derive the complexity of three aspects:

- *Forward pass*: the forward pass represents the prediction of a NN regarding input samples and the associated loss,
- *Backward pass*: the backward pass allows to backpropagate the errors across the NN and update its weights,
- *Weights update*: updating the weights of a NN often relies on optimizers such as GD, SGD, Adam, etc.

Regarding the forward pass, the complexity is the one from the LCDPDNN. Moreover, we use a quadratic loss function, which complexity is simply 2FLOPs. Thus, the forward pass complexity is simply 144 FLOPs/symbol.

About the backward pass, the complexity is higher because the backpropagation algorithm is used to compute the gradients of the weights. From the backpropagation equations presented in [67], we can express the complexity of this algorithm. Consequently, one must evaluate the derivatives of the loss regarding all the weights starting from the output of the **NN** to get the overall complexity. Based on [68] and [69], the number of **FLOPs** of the backward pass is approximately 2 times the one of the forward pass. Thus, the complexity of the backward pass is 288 **FLOPs**/symbol.

Finally, concerning the weights update, we are using the Adam optimizer [53]. From the equations of the optimizer, we can easily count the number of **FLOPs** leading to 22 **FLOPs**/weight/step.

Since we perform meta-learning on **AM/AM NN DPD**, we only consider the **FLOPs** of the latter architecture. Based on equation (3.9), the above assumptions, 22 weights and 13 gradient steps, the total number of **FLOPs** is given by,

$$P_{\text{MAML}} = B_{\text{DPD}} \underbrace{(3(6N_\rho + 42))}_{\text{forward+backward}} + \underbrace{6292}_{\text{Adam}} \quad (4.9)$$

Considering that **MAML** needs 3000 **IQ** symbols to converge, the amount of **FLOPs** during an online usage is ~ 708 k**FLOPs**.

4.3 Efficient Weights Selection for Adaptive DPD

Alternatively to **MAML**, we propose another approach to enable adaptive behavior to our proposed **DPD**. This method is based on a weight selector. The goal of this solution is to correctly select specific **DPD** weights to adapt efficiently to a new **PA** state. Indeed, as previously said, a **PA** is subject to many time-varying effects such as the temperature, the input impedance or other electrical variations. For this scenario, to provide an effective **DPD**, the objective is to map a given known **PA** characteristic based on signal measurements. Thus, we introduce the **DPD Neural Network Weights Selector (DPDNNWS)** algorithm.

4.3.1 DPD Neural Network Weights Selector (DPDNNWS)

The **DPDNNWS** approach relies on two main steps:

- *Database of PAs*: multiple **PA** characteristics must be known to provide an efficient adaptive **DPD**,
- *Identification of the new PA state*: based on known **PA** characteristics, we realize a **GLRT** [70] to identify which known **PA** characteristic better suits the current **PA** state.

About the database of **PAs**, we assume having multiple known characteristics denoted by PA_k . The latter can be acquired in laboratory using a test bench, *e.g.* we realize k acquisitions of a **PA** considering time-varying parameters such as the temperature, the impedance, *etc.* For each acquisition, we perform a training of our **LCDPDNN**, and we store the resulting weights $\theta_{\rho,k}$ and $\theta_{\phi,k}$.

Then, considering the system model described in Section 2.2, we assume receiving an unknown signal \mathbf{z} characterized by an a priori unknown **PA** characteristics at the transmitter. The received signal is corrupted by a multiplicative complex coefficient and an additive noise. To choose which known characteristic better suits the current **PA** state, we perform a **GLRT** using the following hypotheses:

$$H_k : \mathbf{z} = h\mathbf{y}_k + \mathbf{n}, \forall k \in E \quad (4.10)$$

where \mathbf{y}_k and E respectively denote the output vector the PA_k characteristic, and the set of known **PA** characteristics. Moreover, $\mathbf{z}, \mathbf{y}_k, \mathbf{n} \in \mathbb{C}^{1 \times N}$.

Remark 10. *It shall be reminded that each \mathbf{y}_k is acquired in laboratory using a test bench.*

The noise vector $n \in \mathbb{C}^{1,N}$ follows a Gaussian sampling distribution. Thus, the probability density function of \mathbf{z} , under hypothesis H_k given h and σ^2 , is given by

$$p_k(\mathbf{z}|h, \sigma^2) = \left(\frac{1}{2\pi\sigma^2} \right)^{N/2} e^{-\frac{1}{2\sigma^2} \|\mathbf{z} - h\mathbf{y}_k\|^2}, \quad (4.11)$$

where $\|\mathbf{u}\|^2$ is the norm vector $\|\mathbf{u}\|^2 = \mathbf{u}\mathbf{u}^H$.

Then, one must determine h and σ^2 to perform the **GLRT**. Under hypothesis H_k , \hat{h}_k and $\hat{\sigma}_k^2$ can be estimated using Maximum Likelihood Estimation (**MLE**):

$$\frac{\partial \log p_k(\mathbf{z}|h_k, \sigma_k^2)}{\partial h_k} = 0, \quad (4.12)$$

$$\frac{\partial \log p_{Hk}(\mathbf{z}|h_k, \sigma_k^2)}{\partial \sigma_k^2} = 0. \quad (4.13)$$

One can show that using derivatives (4.12) and (4.13) respectively, \hat{h}_k and $\hat{\sigma}_k^2$ are given by,

$$\begin{aligned} \hat{h}_k &= (\|\mathbf{y}_k\|^2)^{-1} \mathbf{z}\mathbf{y}_k^H, \\ \hat{\sigma}_k^2 &= \frac{1}{N} \left\| \mathbf{z} - \hat{h}_k \mathbf{y}_k \right\|^2. \end{aligned} \quad (4.14)$$

Finally, by replacing the estimators \hat{h}_k and $\hat{\sigma}_k^2$ in equation (4.11), we can define the following rule:

$$k_{opt} = \arg \max_k (\|\mathbf{y}_k\|^2)^{-1} \left\| \mathbf{z}\mathbf{y}_k^H \right\|^2, \quad (4.15)$$

with k_{opt} defining the index of the closest **PA** characteristic \mathbf{PA}_k w.r.t. to the current **PA** state. Basically, the optimal decoder is based on a correlation of the received symbols by a set of known signals.

Now, we can map the k_{opt} index to previously stored **NN** based **DPD** weights. Then, we can load the weights $\theta_{\rho, k_{opt}}$ and $\theta_{\Phi, k_{opt}}$ in the **DPD**. Thus, the index k_{opt} allows selecting the best **DPD**.

4.3.2 Numerical simulations

To assess the performance of the **DPDNNWS** algorithm, we choose the same **PA** model as in Section 4.2.3. We also consider the variation of the parameter p . Specifically, we consider having k **PA** characteristics leading to $p_k = 0.7 + k\delta_p$, $k \in \mathbb{N} \mid 0 \leq k \leq \frac{1.5-0.7}{\delta_p}$, where δ_p corresponds to a knee factor step. For each p_k , we have a set of corresponding weights which lead to near-optimal **DPD**. Finally, the complex coefficient h is chosen randomly according to a uniform distribution.

4.3.2.1 EVM performance

First, we show the **EVM** performance of the **DPDNNWS** approach regarding a wide range of **PA** characteristics. Thus, in Figure 4.5, we present the performance of the **DPDNNWS**, with $p \in [0.7, 0.025, 1.5]$ represented by the candidate “knee factor”. These candidates denote the current state of the **PA**, operated with an $IBO_{dB} = 8\text{dB}$. Moreover, we consider knowing 9 **PA** characteristics with $\delta_p = 0.1$. The “limiter”, i.e. the **PA** linear until saturation, is represented in green while the current **PA** state is pictured in blue. The brown curve represents the performance of the system using the **DPDNNWS**. Gray squares correspond to a state of the current **PA** correctly mapped to a known characteristic leading to near-optimal **DPD**. The performance of the **DPDNNWS** algorithm is the same as the **LCDPDNN**

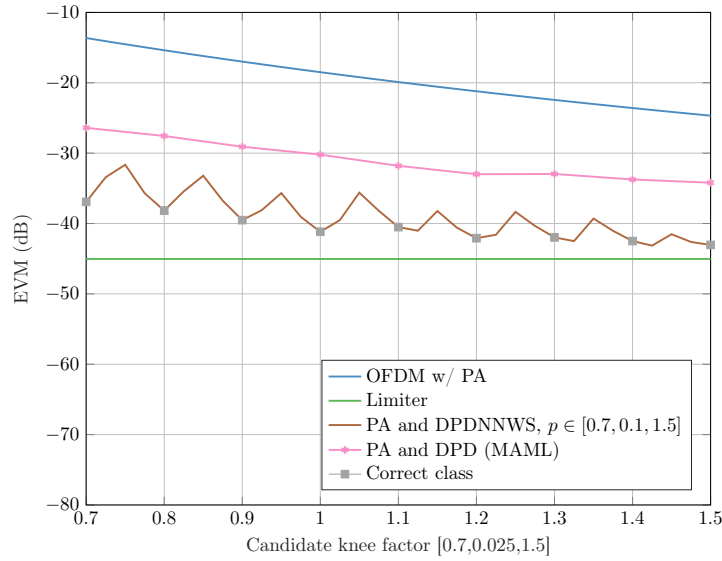


Figure 4.5: EVM performance in (dB) of DPDNNWS and MAML, $IBO_{dB} = 8\text{dB}$

only considering the gray squares. However, we can denote up to 8dB loss of performance when the DPD is not well-chosen by our solution. Then, this algorithm is very sensitive even if the variation of p is small. This means that the number of known characteristics shall be chosen carefully. Besides, to perform an accurate selection of weights, only 10^3 IQ symbols are required. Moreover, we compare the DPDNNWS to MAML in terms of EVM. We can see that the selector approach performs better than meta-learning even when the weights are not correctly chosen.

Next, in Figure 4.6, we evaluate the EVM performance of the weights selector regarding the number of known PA characteristics. The current PA characteristic is changing according to $p = 0.7 + 0.05j, j \in \mathbb{N} \mid 0 \leq j \leq 16$. The red dots on the figure correspond to the PA characteristic selection of the DPDNNWS, triggering the use corresponding weights for the DPD. As an example, taking the first column of the matrix on Figure 4.6, we only have one a priori know PA characteristic with $p = 1.5$. Then, the selector will always choose this characteristic and by extent the associated weights to linearize a PA with $p = 1.5$.

We observe a severe performance degradation when $k \leq 6$. Indeed, the DPD cannot linearize correctly the PA when the DPDNNWS fails to select the correct weights. Moreover, this performance loss is stronger on low values of p , i.e. nonlinear PA states, which is undesirable. However, if we choose $k > 6$, the degradation starts fading, and we notice more homogenous performance across all PA states. Eventually, the last column is the achievable limit of our algorithm considering that we know every single state of the PA. This is not feasible in practice since we may have an infinity of PA states.

4.3.2.2 Spectral analysis

To evaluate the level of OOB distortions, we choose a PA with $p = 0.7$, operated with an $IBO_{dB} = 8\text{dB}$. Regarding the analysis conducted on the achievable EVM using our proposed approach, we consider a DPDNNWS with 9 a priori acquired characteristics. We also add a penalty to the DPDNNWS, i.e. we consider that it maps the considered PA to a known characteristic with $p = 0.75$. Figure 4.7 presents the PSD of the DPDNNWS compared to the use LCDPDNN adapted to the considered PA. We can see that both techniques exhibit the same performance. It means that with only 9 known PA characteristics, the DPDNNWS allows to successfully correct the OOB distortions.

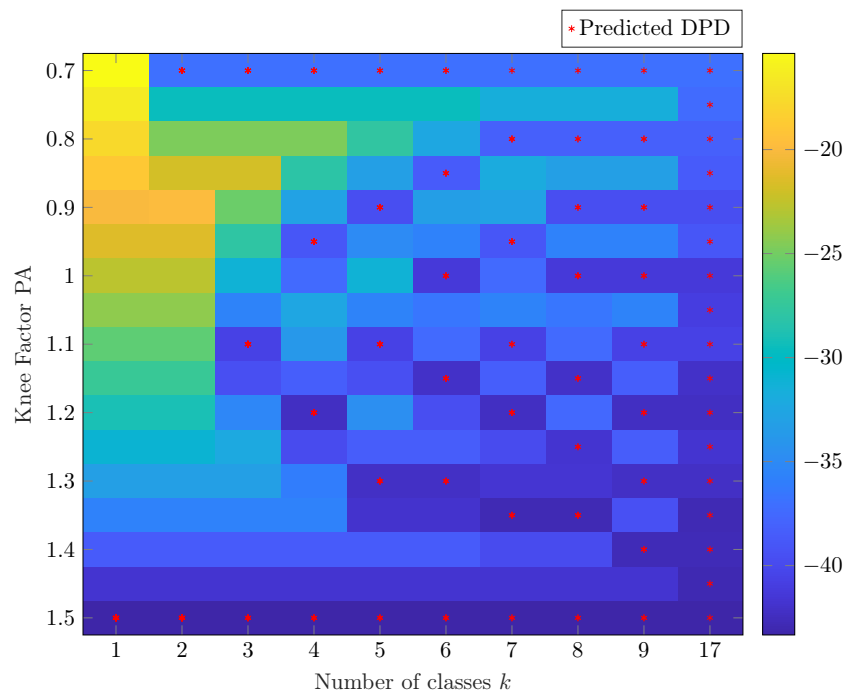


Figure 4.6: EVM performance in (dB) using DPD chosen by the DPDNNWS, $IBO = 8\text{dB}$

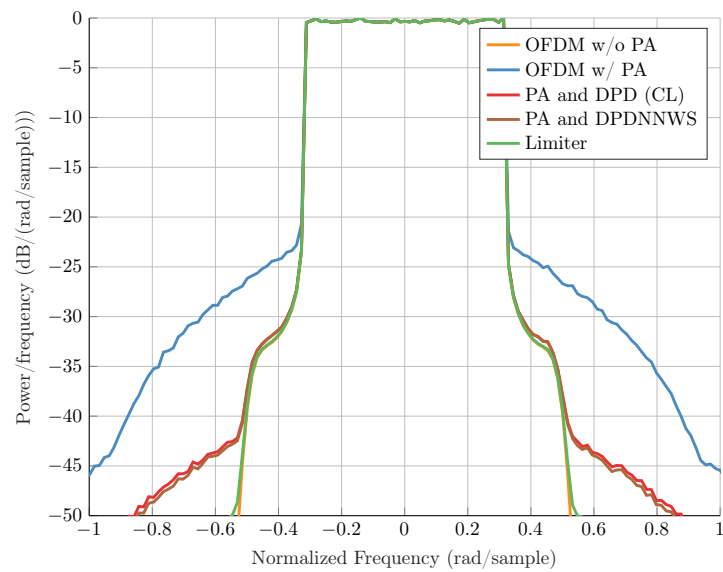


Figure 4.7: PSD using DPD (CL) and DPDNNWS with $p = 0.7$, $IBO_{dB} = 8\text{dB}$

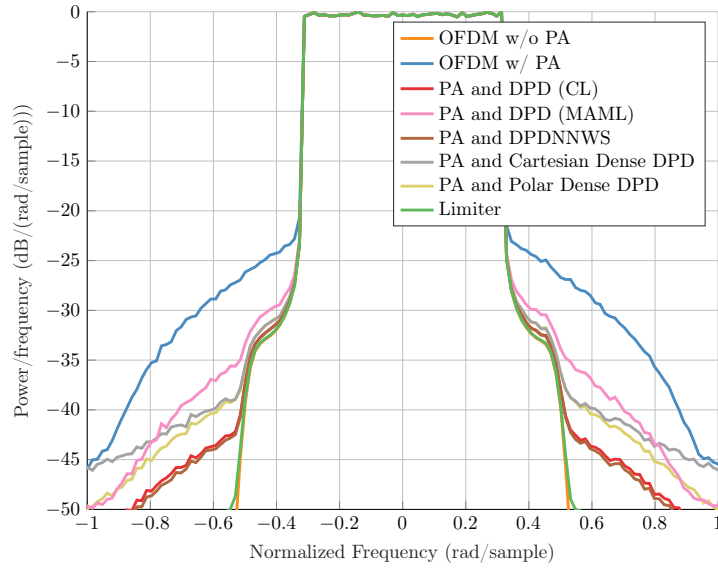


Figure 4.8: PSD comparison using presented algorithms and literature solutions with $IBO = 8\text{dB}$, $p = 0.7$

4.3.3 Complexity

Similarly to other proposed algorithms, we can also calculate the total number of FLOPs required to execute the DPDNNWS algorithm. Based on equation (4.15), we can derive the number of FLOPs for each matrix operation. The squared norm vector and complex conjugate consumes respectively, $3B_{DPD}$ FLOPs and B_{DPD} FLOP where B_{DPD} denotes the size of the vector. Finally, the arg max counts for N_E FLOPs since it performs a comparison. N_E represents the number of known PA characteristic. Thus, based on these values the number of FLOPs is given by,

$$P_{DPDNNWS} = 8B_{DPD}N_E. \quad (4.16)$$

In addition, using previously defined simulation parameters, we have $B_{DPD} = 1000$ and $N_E = 9$. Thus, DPDNNWS approach consumes 72 kFLOPs to select the correct DPD.

4.4 Comparison between DPD algorithms

In this section, we propose an extensive comparison between each of our proposed algorithm to perform DPD for PA. Namely, we compare LCDPDNN, MAML, DPDNNWS and state-of-the-art algorithms proposed in [43] and [44].

4.4.1 Spectral analysis and ACLR

We analyze here the ability of each DPD algorithm to mitigate the OOB distortions induced by the PA. Particularly, Figure 4.8 presents the PSD at the output of the PA using each DPD algorithm. We can notice that every technique offers a significant spectral degrowth of the PA nonlinearities. The use of DPDNNWS allows achieving the same performance as the “conventional” learning. Moreover, MAML leads to lessen performance due to its generalization which needs further improvement. Eventually, state-of-the-art algorithms reach more or less the same performance as MAML. MAML still presents some “visible” spectral regrowth even if it reaches a lower noise than the “Cartesian Dense DPD”. Thus, we can state that every algorithm can successfully linearize the PA, and our proposed solutions demonstrate a better performance than the ones from the literature.

Table 4.2: ACLR in (dB)

IBO	PA	Limiter	LCDPDNN	DPDNNWS	MAML	Cartesian [43]	Polar [44]
4dB	-29	-34	-33	-33	-33	-32	-33
6dB	-31	-41	-41	-39	-39	-39	-40
8dB	-33	-53	-47	-47	-42	-42	-44

Based on this PSD, we evaluate the ACLR of each PSD for multiple IBO configurations. Table 4.2 presents the ACLR using each DPD technique when $IBO_{dB} = 4\text{dB}$, $IBO_{dB} = 6\text{dB}$ and $IBO_{dB} = 8\text{dB}$. First, we can notice that when $IBO_{dB} \leq 8$, all techniques have roughly the same ACLR. This phenomenon is due to the fact that a lower IBO significantly increases the PA level of nonlinearities. Thus, it becomes harder for the DPD to linearize the PA. This explains why performance of each DPD technique become closer when the IBO decreases. However, when $IBO_{dB} = 8\text{dB}$, we can clearly see the difference between each algorithm. MAML exhibits a higher ACLR than other proposed solution, but is still close to literature solutions, in particular [43].

4.4.2 Synthesis on all performance metrics

In this paragraph, we present an extensive comparison of our presented algorithms and multiple works around DPD to assess the viability of our work. Table 4.3 provides multiple criteria to evaluate the efficiency of each DPD algorithm. Specifically, the ACLR gain denotes the difference between the ACLR using the PA only and the ACLR using the DPD. Some information about literature works cannot be found, and are then marked as “N/S”, i.e. “Not specified”.

First, it can be said that our proposals exhibit lower complexity than related works. This criterion is very important if we want to implement the solution on a hardware platform. Indeed, complexity has a direct impact on latency and thus the online usage of the DPD. Related works have higher complexity due to their “blackbox” design, discussed in Chapter 3. Second, MAML and DPDNNWS provide an adaptive behavior contrary to other solutions. Remarkably, the latter solutions respectively consume 3×10^3 and 10^3 IQ symbols, i.e. 1 to 3 OFDM symbols. Then, our adaptive algorithms are suited for real-time applications. This is not the case for other solutions that do not take the adaptivity explicitly into account. We can see that solutions from [46], [43] and [44] consume a lot of data to adapt the DPD weights because a full training is required. Interestingly, the solution presented in [45] consume a fair amount of data. Nevertheless, the derived PA model is static, and requires a new training each time the PA is changing. This induces more latency which is not suitable for online usage.

Finally, regarding EVM and ACLR, our proposals give better results even if MAML still presents lower performance compared to the DPDNNWS. In [46], authors proposed a high-performance DPD, but the complexity of their algorithm is challenging, approximately three times the complexity of our LCDPDNN which is prohibitive in real-time scenarios.

4.5 Conclusion

In this chapter, we proposed two different approaches to perform both low-complexity and adaptive DPD at a low cost. First, we explored the lead of meta-learning and especially the use of the MAML algorithm. This algorithm allows finding the best weights initialization for our NN based DPD, allowing to adapt quickly the DPD regarding a change in the PA characteristic. This solution consumes few data to perform adaptation during online scenario resulting in low-latency. It achieves fair performance compared to “conventional” learning. Differently, we proposed another algorithm performing a selection of the best

Table 4.3: Comparison with state-of-the-art

Related Work	ACLR Gain	EVM	FLOPs	Online	Adaptive (IQ Symbols)
RT2DNN [46]	12dB	N/S	412	No	Poor ($\approx 10^6$)
DPD w/o ILA [45]	N/S	N/S	184	No	Poor ($\approx 10^4$)
Cartesian Dense DPD [43]	9dB	-34dB	322	No	Poor ($\approx 10^6$)
Polar Dense DPD [44]	11dB	-34dB	160	No	Poor ($\approx 10^6$)
Our work					
LCDPDNN (CL)	14dB	-38dB	142	No	Poor ($\approx 3 \times 10^5$)
LCDPDNN (MAML)	8dB	-29dB	142	Yes	Good ($\approx 3 \times 10^3$)
LCDPDNN + DPDNNWS	14dB	-37dB	142 + 72	Yes	Excellent ($\approx 10^3$)

weights for the DPD. The DPDNNWS relies on hypotheses tests with a priori known PA characteristics, and pretrained DPD weights. This proposal brings out a real benefit in terms of complexity and data usage during online scenarios. Ultimately, we showed that our proposed approaches allow adding an adaptive behavior to the developed LCDPDNN with few data and low-latency during the calibration process.

Notwithstanding, MAML still needs some improvements to perform better. New works have emerged trying to improve MAML, e.g. the work in [71] uses implicit differentiation to improve even more the generalization of the algorithm and lower its computing resources. Nevertheless, even with efficient solutions to provide low-complexity and adaptive DPD, a main issue is still challenging. Implementing our solution on hardware is not possible without some adjustments. Indeed, since our DPD is based on NNs, it then exhibits high-cost operations in terms of latency and energy. By default, NNs operations are performed using floating-points number coded over 32-bits; the precision is then very high. However, for efficient hardware systems, this quantization scheme is unacceptable. Then, the pending questions to solve are: *can we lower the quantization of the DPD?*, *how can we cope with the underlying quantization effects?* and *what are the benefits of such action?*.

Thus, in the next chapter, we investigate quantization of NNs and its effects to answer the aforementioned concerns. This implies, finding a way to cope with quantization noise, and assess the benefits in terms of resource usage and energy consumption.

The technical content of this chapter has been approved by the following patent and journal paper:

[P1] A. Falempin, R. Zayani and J-B. Doré, “Device for linearizing a power amplifier of a communication system by digital predistortion,” Issued in July 05, 2022, US2107230.

[J1] A. Falempin, J-B. Doré, R. Zayani and E. Calvanese Strinati, “Low-Complexity Adaptive DPD: From online optimization to meta-learning,” IEEE Transactions on Broadcasting, pp:pp, 2022.

Quantized Neural Network based DPD for Efficient Hardware Implementation

Contents

5.1	Introduction	59
5.1.1	Motivations	59
5.1.2	Related Work	59
5.1.3	Contributions	60
5.2	Quantized Neural Network based DPD	60
5.2.1	Number representation	60
5.2.2	Post-Training Quantization (PTQ)	62
5.2.3	Quantization-Aware Training (QAT)	63
5.2.4	Numerical simulations	64
5.3	Towards FPGA implementation	68
5.3.1	FPGA architecture and programming	68
5.3.2	Numerical simulations	69
5.4	Conclusion	71

5.1 Introduction

To improve the energy-efficiency and break down the complexity of Radio-Frequency (RF) wireless transmission chains, this chapter investigates the quantization of Neural Networks (NNs) based Digital Pre-Distortions (DPDs) for cost-effective hardware implementation. Usually, NNs are performing floating-point operations [72] using 32-bits. This default quantization scheme is often neglected by the community as many researchers seek the performance rather than the complexity and energy-efficiency. However, future wireless communications networks must offer sustainable designs of every component and more particularly the transceivers designs. Thus, it becomes increasingly important to lower the quantization of NNs in order to reduce their latency and power consumption. To that extent, we propose to perform quantization on the NN based DPD proposed in Chapter 3. More specifically, in this chapter, we explore the effects of applying low-bit quantization to our NN based DPD. Then, we use a Quantization-Aware Training (QAT) on our DPD to mitigate the quantization noise encountered while quantizing the NNs. In addition, we provide an estimation of computing resources using a FPGA synthesis.

5.1.1 Motivations

NNs often suffer from high computing resources usage leading to high-latency operations, and high power consumption mainly caused by high precision quantization scheme, *i.e.* 32-bits. Concerning the DPD for PA, the final goal is to increase the overall energy-efficiency of the transmitter chain while keeping low-latency and linearity of the transmission. Thus, as is, our proposed Low-Complexity DPD Neural Network (LCDPDNN) does not really meet these requirements. Besides, since the literature often considers NN as a “blackbox”, the quantization aspect is often neglected leading to poor energy-efficiency and increased computational overhead.

Thus, our objective is to lower the number of bits required to perform DPD using our LCDPDNN. By applying low-bit quantization, we ensure having an energy-efficient solution. However, one must cope with the quantization noise to avoid inoperable DPD. Then the problem here is twofold: i) a low-bit quantization is desired to minimize the power consumption of the DPD, ii) quantization noise must be cancelled to keep the same performance as high-bit quantization.

5.1.2 Related Work

As shown multiple times in the previous chapters of this thesis, DPD has been widely investigated since decades. NNs have shown to be promising at performing DPD for PA linearization as stated in Chapter 3. However, most of the works on this topic do not consider implementation of their solution. For instance, in [73], the authors proposed an experimental setup of their NN based DPD. However, this implementation considers that the DPD runs on a computer. Thus, this work does not consider the aspects we are trying to tackle here, namely power consumption and low-bit quantization for hardware implementation. Similarly, in [45] and [74], authors consider using a “weblab” to acquire PA characteristics [75], and then run their DPD training on a software such as MATLAB or Python. While this shows the ability of their DPD to linearize real PA, the fact remains that hardware complexity and latency are still challenging here.

Accordingly, to comply with power constraint and latency, quantization of NNs has been considered to improve the implementability of the latter on embedded systems, which are limited in terms of resources. This research topic is then highly active in the literature since it greatly optimize resource usage and energy consumption [76]. Hence, one must consider applying low-bit quantization for all NN operations. In [77] and [78], authors made an extensive review of different approaches to perform quantization of NNs. In wireless communications community, NNs are becoming essential to solve many nonlinear problems. Nevertheless, few works are interested in quantization. A recent work proposed to jointly optimize precision, energy consumption and accuracy of federated NNs [79]. Authors have

demonstrated all the aforementioned benefits from quantizing **NNs**, but their model relies on a specific on-chip architecture dedicated to run Convolutional Neural Network (**CNN**) for image processing.

Eventually, in [80], the implementation of a **NN** based **DPD** is considered on a **FPGA**. While authors provide a **FPGA** design, they do not take quantization into account resulting in a suboptimal implementation. Thus, we have seen that quantization brings out must have benefits regarding **NN** implementation. However, most works do not take it into account. Specifically, there is actually no work that jointly approach **NN** quantization and hardware design for implementation.

5.1.3 Contributions

The contributions of this chapter are twofold:

- **Quantization of LCDPDNN**: we propose to enable quantization of our developed **LCDPDNN**, –see Chapter 3, to enhance the resource usage of the transmission chain and by extent its energy efficiency. We propose to use a fixed-point quantization on weights and outputs of the **NN**. To do so, we use Post-Training Quantization (**PTQ**) and **QAT** to appreciate quantization effects and correct them.
- **Implementation on FPGA**: we propose an implementation of the **LCDPDNN** on a **FPGA** considering the same quantization. Doing so, we are able to provide an estimation of the resource usage. A **FPGA** design is then proposed taking into account the quantization aspects.

The technical content of this paper is based on conference paper [81]. A journal paper is currently under preparation to provide additional elements to [81]. The remainder of this chapter is organized as follows. Section 5.2 presents approaches to enable low-bit quantization for our designed **LCDPDNN** without significant performance loss. Numerical evaluation is conducted to assess the benefits of quantizing our **NNs**. Then, Section 5.3 provides a **FPGA** design of our **DPD** solution to estimate the overall resource usage and lead the path towards a functional hardware implementation of our **DPD**.

5.2 Quantized Neural Network based DPD

We present here our approach to quantize efficiently the **LCDPDNN**. Namely, we first present the quantization scheme used for our **LCDPDNN**. Then, we give two approaches to perform quantization with and without compensation of the noise brought by the quantization operation.

5.2.1 Number representation

A fractional number can be represented using two different ways, floating-point arithmetic and fixed-point arithmetic. We give some practical elements on each arithmetic.

5.2.1.1 Floating-point arithmetic

It exists multiple ways to perform floating-point arithmetic. However, the most common representation is the one from the *IEEE-754* standard [72]. Let's suppose having a number coded on 32-bits. According to [72], the representation of the number is divided in three parts:

- *Sign*: the sign is actually coded using 1 bit. When the value of this bit equals 1 the number is negative, and positive otherwise;
- *Mantissa*: it is coded on 23 bits and represents the concatenation of integer part and fractional parts of the fractional number, omitting the most significant bit;
- *Exponent*: it is coded on 8 bits and represents the position of the radix point.

As an example, let's represent the number 85.125 using floating-point 32-bits. The process can be divided in several steps:

Step 1: Define the sign bit, here it is 1 since we have a positive number.

Step 2: Express the integer and fractional parts of the number in binary. For the above number, we have $85_{10} \sim 1010101_2$ and $0.125_{10} \sim 001_2$. Thus, $85.125_{10} \sim 1010101.001_2$

Step 3: Rewrite the binary number using binary scientific notation, we move the radix point to the first bit with the value 1. The number of radix movements gives the scientific notation. Here, we then have 1.010101001×2^6

Step 4: To get the exponent part of the number on 32-bits, we convert $127 + 6 = 133$ in binary. It gives 10000101

Step 5: The mantissa is the binary digits of the binary scientific notation, omitting the most significant bit. Then, the mantissa is 010101001. Since it has to be coded on 23-bits, we put the remaining bits at zero.

Thus, in floating-point 32-bits, 85.125 becomes:

$$\underbrace{0}_{\text{Sign}} \underbrace{10000101}_{\text{Exponent}} \underbrace{010101001000000000000000}_{\text{Mantissa}}$$

Remark 11. Some numbers have a special representation in terms of exponent and mantissa.

Remark 12. The minimum number in floating point is approximately $-3.4 \cdot 10^{38}$ and the maximum value is $3.4 \cdot 10^{38}$. There are many rules in the IEEE-754 standard that define more specific ranges regarding the number.

Besides, one can perform a quantization using floating-point by changing the number of bits for the exponent and/or mantissa.

5.2.1.2 Fixed-point arithmetic

For this representation, there is no standardization as in floating-point arithmetic. A fractional number is represented by its integer and fractional parts. We denote by $S(i, f)$ a signed fixed point representation with i bits for the integer part including the sign bit and f for the fractional part. Then, a decimal number is represented by its binary form when using fixed-point.

As an example, let's take again the number 85.125. Fixed-point representation of this number corresponds to the step 2 of the floating-point conversion in the above paragraph. Thus, our number becomes:

$$\underbrace{0}_{\text{Sign}} \underbrace{1010101}_{\text{Integer}} . \underbrace{001}_{\text{Fractional}} \quad (5.1)$$

This number is represented using the scheme $S(8, 3)$.

Remark 13. Besides, considering the scheme $S(i, f)$, the range of the number r represented using $N_b = i + f$ bits is $[-2^{i-1}, 2^{i-1} - 2^{-f}]$ with a step of 2^{-f} . Moreover, for a fixed value of N_b , the higher i , the wider range of numbers and the lessen precision. Conversely, the higher f , the higher

Table 5.1: Comparison between floating-point and fixed-point

	Precision	Range	Resource usage
Floating-point	Small values	Higher	Higher
Fixed-point	Uniform	Smaller	Smaller

precision and the smaller range.

As a toy example, let suppose having $S(1, 1)$ to represent a number r . Then $r \in -1, -0.5, 0, 0.5$.

Now, suppose that we want to represent a decimal number r using a fixed-point scheme $S(i, f)$. The resulting fixed-point number is given by,

$$Q(r, i, f) = \text{clip}\left(\frac{\lfloor r2^f \rfloor}{2^f}, \left[-2^{i-1}, 2^{i-1} - 2^{-f}\right]\right), \quad (5.2)$$

where $\lfloor \cdot \rfloor$ performs rounding half to even and $\text{clip}(u, [min, max])$ returns min if $u < min$, max if $u > max$ and u otherwise. We denote $N_b = i + f$, the total number of bits for the considered scheme. Hence, one can act on the quantization of a number by changing i or f .

5.2.1.3 Floating-point v.s. fixed-point

Table 5.1 presents a qualitative comparison between both quantization schemes. Floating-point representation is much more precise for value around 0. However, the precision begins to decrease when the value rises. Considering fixed-point, we have a uniform range of values, but the range is smaller. Yet, fixed-point is easier to implement, and consumes less resource than floating-point. In addition, fixed-point number can be stored as integers. Then, it is more suited to embedded processing units, e.g. **FPGA**.

Important note 6. *Thus, in this chapter, we will only consider the use of fixed-point representation and associated quantization rule (5.2). Indeed, this will allow less resource usage, a requirement to fulfill.*

5.2.2 Post-Training Quantization (PTQ)

To perform quantization on a **NN**, we can use a simple technique named **PTQ**. **PTQ** applies quantization on the weights and neurons operations of the **NN**. As indicated by its name, **PTQ** requires a pre-trained **NN**. The goal of this type of quantization is to evaluate the impact of the quantization over the considered **NN**.

Considering our **LCDPDNN**, we have to apply fixed-point quantization on every parameter, and output produced by each operation. Thus, the quantized version of the **NN** becomes:

For **AM/AM NN** based **DPD**,

$$\begin{aligned} \widehat{f}_\rho^{-1}(u) &= Q[\text{ReLU}(Q(g_0(u)) + Q(g_1(u)))], \\ \text{where } \begin{cases} g_0(u) &= \gamma_0(u) + (1 - \gamma_0(u))u, \\ \gamma_0(u) &= Q\left(\left(1 + e^{-Q(\alpha)(u - Q(\omega_\rho))}\right)^{-1}\right), \\ g_1(u) &= \sum_{j=1}^{N_\rho} Q(\omega_j^\rho) [\text{ReLU}(Q(\mathbf{W}_\rho)(u - Q(\omega_\rho)) + Q(\mathbf{b}_\rho))]_j + Q(\beta_\rho), \end{cases} \end{aligned} \quad (5.3)$$

For **AM/PM NN** based **DPD**,

$$\begin{aligned} \widehat{f}_{-\Phi}(u) &= Q[\gamma_1(u)Q(g_2(u))], \\ \text{where } \begin{cases} \gamma_1(u) &= Q\left(\left(1 + e^{-Q(\beta)(u - Q(\omega_\Phi))}\right)^{-1}\right), \\ g_2(u) &= \sum_{j=1}^{N_\Phi} Q(\omega_j^\Phi) \text{ReLU}[(Q(\mathbf{W}_\Phi)(u - Q(\omega_\Phi)) + Q(\mathbf{b}_\Phi))]_j, \end{cases} \end{aligned} \quad (5.4)$$

The function Q is defined in (5.2). i and f are omitted for simplicity. Besides, these equations are the same as (3.4) and (3.6), except that we apply the Q function to quantize weights, dot products, additions and multiplications. Activation functions, namely sigmoid and ReLU, are also quantized. Moreover, the input u is also quantized. It means that $\widehat{f}_\rho^{-1}(u)$ and $\widehat{f}_{-\Phi}(u)$ must have the same quantization scheme as u .

Using **PTQ** is then straightforward because we only have to apply the function Q to the whole network. Thereafter, we will consider the notations $Q(\text{NN}_\rho)$ and $Q(\text{NN}_\Phi)$ to respectively denote the quantized versions of **AM/AM** and **AM/PM NNs** based **DPD**.

5.2.3 Quantization-Aware Training (QAT)

When using **PTQ**, we apply “aggressive” quantization to our **NN**. This quantization will change the values of the weights, and outputs of the neurons. If a low-bit quantization is applied, the resulting **NN** based **DPD** will not be optimal anymore to perform efficient linearization. Thus, we must find a way to counteract the effect of quantization on our **NNs** to retrieve an optimal **DPD**.

QAT is a solution to cope with the quantization noise brought in the **NN**. Actually, it is a training algorithm that takes into account the effect of quantization. The goal of this algorithm is to train our **NN** to adjust its weights to mitigate the quantization noise. Figure 5.1 presents the **QAT** algorithm for one gradient step. **QAT** is detailed in algorithm 1.

Algorithm 1 QAT

Require: \mathcal{D} : training dataset

Require: η : learning rate for gradient descent

Ensure: θ : pre-trained floating-point 32-bits weights

while not done do

 Quantize the NN during forward pass, $\hat{f} = Q(\text{NN})$

 Evaluate $\nabla \mathcal{L}(\mathcal{D}, Q(\theta^{k-1}))$

 Update θ : $\theta^k = \theta^{k-1} - \eta \nabla \mathcal{L}(\mathcal{D}, Q(\theta^{k-1}))$

 ► Use STE for round function

end while

Thus, this learning algorithm is similar to a “conventional” learning in floating-point. However, some major differences exist. First, we apply quantization only during the forward pass. Thus, the

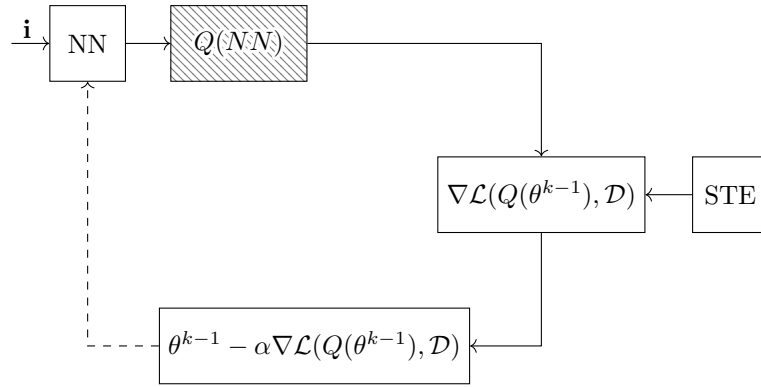


Figure 5.1: QAT algorithm for one gradient step

optimization of the loss takes into account the quantization of weights and activations. The quantization operation uses a round function which is a step function. The derivative of this kind of function is almost zero everywhere. This implies vanishing gradient, *i.e.* zero-valued gradients, leading to convergence issue during GD. To avoid this problem, a common trick is to use a Straight-Through Estimator (STE) [56]. This estimator acts as a bypass of the rounding function during the backward pass. Namely, it considers the round function as the identity function leading to a derivative with a value of 1. While being biased, it solves the vanishing gradient problem without impacting the convergence speed.

Important note 7. Gradients computation are performed using floating-point over 32-bits. The update of the weights is performed by the Adam optimizer using as well floating-point. Doing so also avoids vanishing gradient issues.

Thus, QAT adapts the weights such that after quantization, the NN becomes robust to the selected quantization for the weights. Then, it appears that this algorithm is mandatory, especially for low-bit quantization where we expect high noise.

5.2.4 Numerical simulations

Same as the previous chapters, we consider a PA whose characteristics are defined in Section 2.3.2.2. We consider the same parameters as defined in [30], except $p = 0.7$ to have a more nonlinear PA. Besides, the PA is operated with an $IBO_{dB} = 8\text{dB}$. We evaluate the performance of our quantized DPD using PTQ and QAT w.r.t. to EVM metric and spectral analysis. The physical layer parameters remain the same as in Chapter 3 – see Table 3.3. Besides, the number of neurons for each NN remains unchanged, *i.e.* $N_\rho = 6$ and $N_\Phi = 3$.

5.2.4.1 Choice of quantization schemes

To perform PTQ and QAT, one must carefully choose the quantization schemes regarding the weights and activations. Table 5.2 presents all the quantization schemes for each weight and activation function of the LCDPDNN. About the activation functions, they represent a bottleneck for the NNs in terms of quantization. Indeed, the γ function is a sigmoid which largely contributes to the PA linearization. Besides, the sigmoid ranges between 0 and 1. Yet, we use the scheme $S(0, 12)$ which represent an unsigned fixed-point number. Since the sigmoid produces small values, under 12 bits, the DPD becomes unusable. For the ReLUs, we employ an $S(2, 10)$ scheme because the output may exceed 1. In addition, considering a practical system, Digital-to-Analog Converter (DAC) and Analog-to-Digital Converter (ADC) are operated using a limited number of bits. For our study, we consider DAC and ADC operated using 12 bits. Hence, it motivates the choice of this number of bits for the NNs outputs and activations.

Table 5.2: Quantization of NN weights and activations

Parameters	Quantization scheme
α, β	$S(8, 5)$
b_z	$S(1, f)$
ω_ρ, ω_Φ	$S(1, 8)$
$\mathbf{W}_\rho, \mathbf{W}_\Phi$	$S(3, f)$
$\mathbf{b}_\rho, \mathbf{b}_\Phi$	$S(1, f)$
$\omega_j^\rho, \omega_j^\Phi$	$S(1, f)$
γ_0, γ_1	$S(0, 12)$
ReLU	$S(2, 10)$

Concerning the weights, we mainly act on the fractional bits f . Indeed, the weights $\omega_j^\rho, \omega_j^\Phi \in [-1, 1)$, $\mathbf{b}_\rho, \mathbf{b}_\Phi, b_z \in [-1, 1)$ and $\mathbf{W}_\rho, \mathbf{W}_\Phi \in [-4, 4)$.

Important note 8. *Considering a fixed-point quantization, lowering the number of bits for the integer part will significantly reduce the range of values. Thus, our NN will not be able to provide satisfying results.*

5.2.4.2 EVM performance

In this paragraph, we analyze the impact of fixed-point quantization on the EVM performance. Figure 5.2 presents the EVM performance of the LCDPDNN without quantization, and with PTQ and QAT. The baseline model, i.e. floating-point 32 bits DPD, is represented by a green line and denotes the performance bound. The yellowish and orange bars are respectively the EVM performance of PTQ and QAT. The number of bits corresponds to the variation of f in the quantization scheme $S(1, f)$.

Remark 14. *The number of fractional bits f in the scheme $S(3, f)$ is also varying. However, the total number of bits concerns the scheme $S(1, f)$.*

First, it can be underlined that PTQ exhibits high performance loss. Indeed, below 7 bits, the EVM does not meet the LTE requirements, i.e. -22dB when using 64-QAM. Moreover, below 5 bits, the DPD becomes useless since the EVM performance is lower than -14dB , the EVM with the PA only. Besides, we can see that the EVM reaches a ceiling of approximately -30dB even with high-bit quantization, i.e. 16 bits. This performance is fair but still far from the baseline model. Based on this analysis, we can state that PTQ is not enough to enable a low-bit quantization without harming the efficiency of the DPD.

Second, we notice that using QAT allows to significantly improve the EVM. The results show that we almost reach the baseline model performance. In addition, the EVM is always higher than -35dB . For quantization levels above 7 bits, the use of QAT permits achieving the same performance as the baseline model. Furthermore, for low-bit quantization, below 6 bits, we have up to 30dB gain compared to PTQ. Thus, based on the EVM performance, it turns out that QAT is essential to ensure an efficient linearization of the PA.

5.2.4.3 Spectral analysis

In addition to the EVM performance analysis, we provide a spectral analysis while using the quantized DPD to better assess its performance. Similarly to other Power Spectral Densities (PSDs) presented in

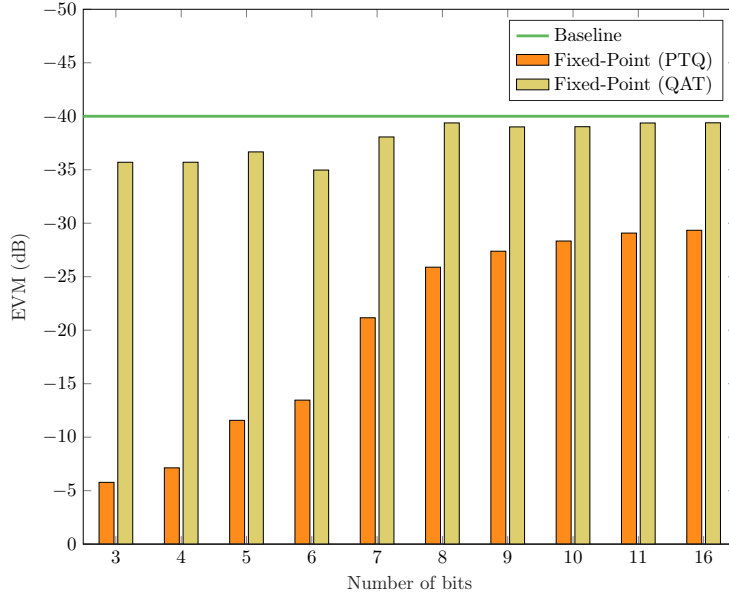


Figure 5.2: EVM performance in (dB) using PTQ and QAT

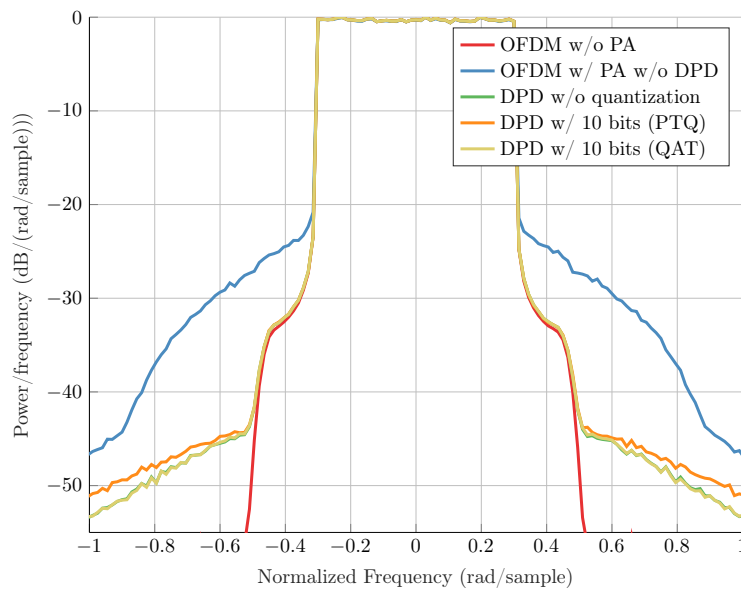
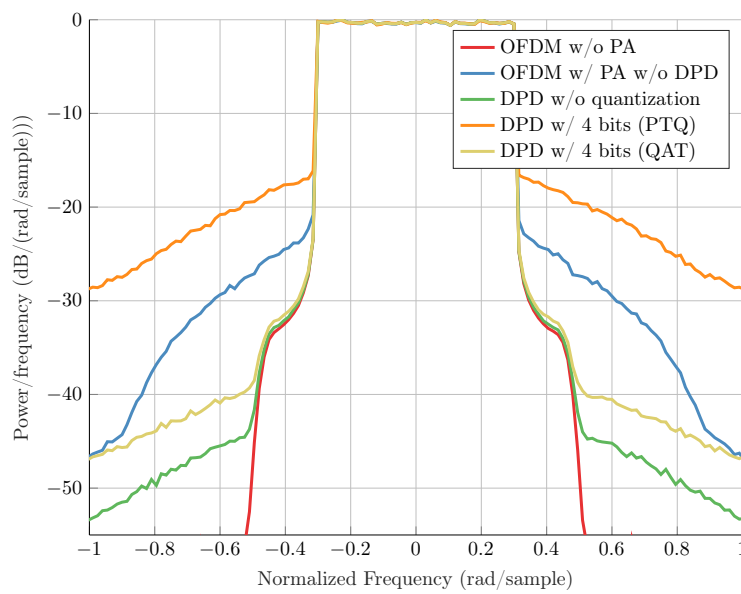
this thesis, we use **DUC** function to design a realistic **RF** transmission chain. Figure 5.3 and Figure 5.4 present **PSDs** using respectively the quantization schemes $S(1, 9)$ and $S(1, 3)$.

On the one hand, in Figure 5.3, we notice that using **PTQ** leads to a small degradation compared to the baseline model, *i.e.* the **DPD** without quantization. This loss is almost negligible since the noise level is below -50 dB. Using **QAT** gives the same performance as the **DPD** coded on 32 bits. Thus, lowering to 10 bits leads to no significant degradation in terms of spectral effects. On the other hand, looking at Figure 5.4, we can see that the **DPD** using **PTQ** brings severe degradation, producing much more spectral regrowth than the **PA** itself. Accordingly, **PTQ** cannot be considered for low-bit quantization. Nevertheless, we can see that the **QAT** calibrates the **DPD** weights, such that the **OOB** distortions exhibited by the **PA** are almost cancelled. Hence, using **QAT**, our **LCDPDNN** successfully learned how to correct the quantization noise using low-bit schemes. Yet, there is still a small spectral regrowth compared to the baseline model. This will not influence on the overall performance because the noise level is below -40 dB, which is sufficient for practical systems. Again, **QAT** proves to be essential when considering a low-bit **DPD** based on **NNs**.

5.2.4.4 Discussion

Through the provided numerical results, we showed the impact of quantization on the **LCDPDNN**, and how to correct its effects. However, some important concerns must be underlined. **PTQ** is the fastest way to provide a quantized **NN**, and shall be then considered when we operate moderate quantization, *i.e.* above 10 bits quantization. Besides, to perform **QAT**, two trainings are required. Converging to an optimal quantized **NN** is complicated considering a training from scratch. Thus, one may consider using the pre-trained **NN** to envisage a **QAT**. Doing so allows adjusting the weights more precisely, and then ensures a satisfying result, close to the baseline model.

Even though **QAT** is more complex than **PTQ**, it is essential if one considers enabling low-bit quantization implementation of **NNs**. For our case, **QAT** can be performed in laboratory. Then, it does not require specific real-time constraints. Eventually, enabling low-bit quantization for our **NNs** opens the path to hardware implementation.

Figure 5.3: PSD using PTQ and QAT, $N_{bits} = 10$ Figure 5.4: PSD using PTQ and QAT, $N_{bits} = 4$

5.3 Towards FPGA implementation

In the first part of this chapter, we successfully demonstrated that low-bit quantization can be enabled for our developed **LCDPDNN**. Specifically, we showed that we are able to mitigate noise brought by quantization, leading to an efficient **DPD**. By doing so, we can consider a hardware design of the **DPD**. We give here a **FPGA** configuration of the **LCDPDNN** that has been evaluated through synthesis. Then, we provide an estimation of the **FPGA** resource consumption regarding the quantization scheme used.

Important note 9. *This work has been conducted in collaboration with an **FPGA** designer. Specifically, the hardware schematics and synthesis have been done by the latter.*

5.3.1 FPGA architecture and programming

We give here some basic insights about **FPGA** for better comprehension. An **FPGA** is a computer chip that can be reconfigured to implement whatever digital hardware circuit. This hardware component is mainly composed of logic blocks that are “programmable”, memory, input/output blocks and clocking [82]. The architecture of an **FPGA** strongly depends on its manufacturer. Thereafter, we consider using a **FPGA** from Xilinx [83]. Specifically, it is composed of:

- Configurable Logic Blocks (**CLBs**) that are often linked together to realize complex functions. The **CLBs** are composed of smaller elements including registers, **LUTs** and multiplexers;
- Block RAM (**BRAM**) which is a type of random access memory embedded on the **FPGA**. It is mainly used to store large amount of data or transfer data using First In First Outs (**FIFOs**);
- A clock driving the speed of data processing;
- “Programmable” wires that are used to interconnect **CLBs** in order to design complex functions;
- Input/output blocks that are used to interface the **FPGA** with external sources.

Remark 15. *The presented architecture elements of the **FPGA** are not directly manipulated while proposing a specific hardware configuration.*

Then, “programming” an **FPGA** relies on a specific process. First, one must specify the schematics of the function to be designed. This allows having a graphical description of the hardware configuration. Second, we translate the design into hardware description language, e.g. Very High Speed Integrated Circuit Hardware Description Language (**VHDL**). Once the **FPGA** configuration has been described, a synthesis is performed using a software from the considered **FPGA** manufacturer. The synthesis permits transforming the **VHDL** into an array of logical gates. The latter corresponds to internal **FPGA** elements such as the **CLBs** and **BRAM**. Eventually, an implementation step will configure the placement and connections of each logical element. Ultimately, a bitstream is generated and transferred into the **FPGA** to configure it.

Remark 16. *The synthesis and implementation stages are automatically made by the corresponding software. In other words, usually, we neither choose where the resources are placed on the **FPGA** nor how they are connected to each other.*

Thus, we apply all this process to design and simulate an **FPGA** configuration of our **LCDPDNN**. Then, we have two configurations, one for the **NN** performing the **AM/AM** correction and the other for the **NN** performing the **AM/PM** correction.

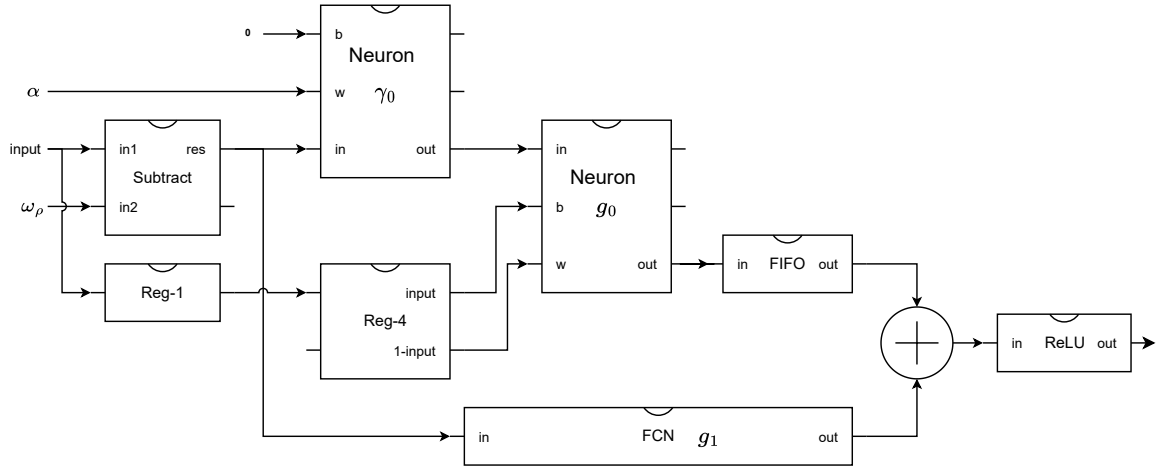


Figure 5.5: FPGA Architecture of NN performing AM/AM DPD

5.3.1.1 FPGA configuration for AM/AM NN based DPD

Figure 5.5 presents the schematic design of the NN performing AM/AM linearization. One can notice that the architecture is different from the one made in Chapter Section 3. Actually, it is a “translation” of the software designed NN in Figure 3.5a into a hardware design.

First, the neuron block is a custom module allowing to realize the neuron operation $f(\mathbf{w}^T \mathbf{x} + b)$ where \mathbf{w} are the weights, \mathbf{x} the vector of inputs, b the bias and f the activation function. The design of this block is skipped for simplicity. Thus, the FCN block models the function g_1 of equation (5.3). Second, the block Reg- n is a succession of registers that keep previous states of the input data, and lasts n clock cycles. Every operation consumes clock cycles. Here, the additions, subtractions and multiplications consume 1 clock cycles. The sigmoid consumes 2 clock cycles because its computation represents the bottleneck of the design. Besides, the sigmoid is approximated using piecewise linear functions.

Then, on the schematic, we can notice that the output of some blocks are aligned. This means that the output of these blocks are to be used at the time. More specifically, the neuron γ_0 requires the input data at the same time as the Reg-4. Since the subtraction costs 1 clock cycle, a register is needed to synchronize the data arrival in Reg-4. Similarly, to obtain the function g_0 , since the neuron sigmoid lasts 4 clock cycles, we have to pipe the input data into Reg-4 for synchronization matter. Next, a FIFO is used to wait for the computation of the FCN block. Finally, an adder followed by a ReLU function will produce the output \hat{f}_ρ .

5.3.1.2 FPGA configuration for AM/PM NN based DPD

Similarly to the conception of the AM/AM NN based DPD, Figure 5.5 presents the design of the NN performing AM/PM linearization. Here, the schematic is much simpler since the NN performing phase shift correction is less complex than the one performing AM/AM correction. Actually, the hardware design is close to Figure 3.5a. The subtraction block gives the quantity $input - \omega_\Phi$ which is the input of the neuron modeling the function γ_1 , and the input of the FCN g_2 . The FIFO serves the same purpose as in Figure 5.5, i.e. waiting for g_2 output. Finally, γ_1 and g_2 are multiplied to get $\hat{f}_{-\Phi}$.

5.3.2 Numerical simulations

Using the above designs, we conduct an analysis of the resource consumption using a FPGA synthesis for further implementation on system on-chip from Xilinx [83]. This evaluation is conducted regarding

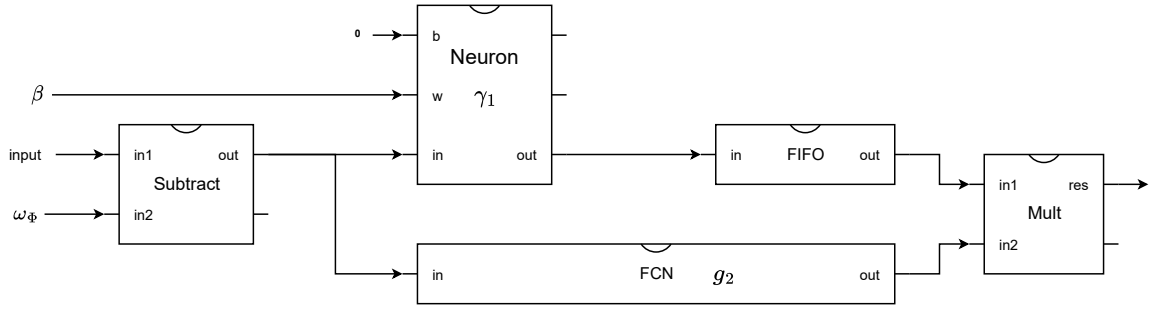


Figure 5.6: Architecture of NN performing AM/PM DPD

the quantization of the NNs defined in Section 5.2. The resource usage of the FPGA is given in terms of CLBs.

Figure 5.7 presents the resource usage of the FPGA for the quantization scheme $S(1, f)$, –see Table 5.2. The resource usage is presented in terms of LUTs performing logical functions, DSP units mainly performing multiplication and accumulation operations, and registers. On Figure 5.7, these three resources are respectively pictured in blue, orange and red. We observe that the resource usage is roughly the same between 6 and 12 bits. Still, the registers' usage lowers a bit when we go from 12 bits to 6 bits. Besides, when the number of bits equals 4, we observe an increase of LUTs and registers usage compared to other quantization schemes. This increase is due to significant decrease of DSP units, approximately 6 times less than the other schemes below 32 bits. Moreover, as a comparison, the implementation of a 4096 points FFT requires 3160 LUTs, 6230 registers and 48 DSPs. Therefore, the proposed FPGA configuration for the LCDPDNN consumes much less resources.

Finally, we can state that 4-bits quantization leads to a significant improvement in terms of resource usage compared to a 32-bits implementation. Moreover, we only have a slight degradation on the performance metrics, *i.e.* EVM and OOB distortions measured on the PSD.

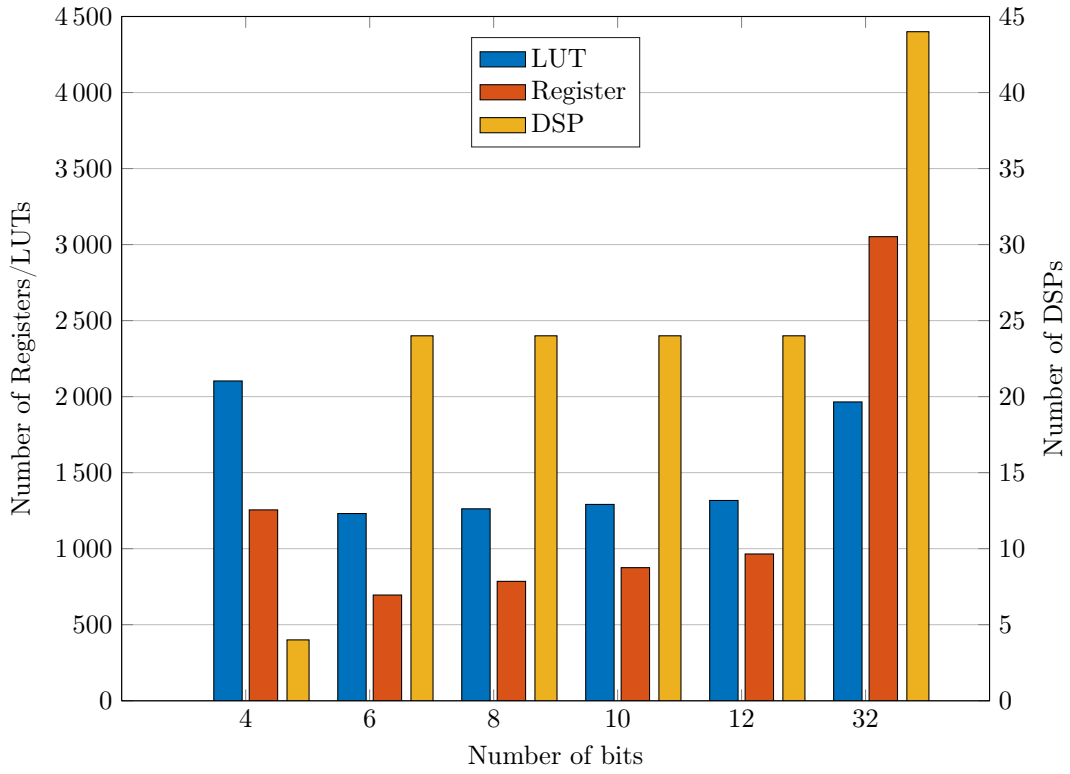


Figure 5.7: FPGA resource consumption

5.4 Conclusion

In this chapter, we proposed an analysis of the impact of quantization over our developed LCDPDNN. Using PTQ, we showed that the quantization operation is leading to unusable DPD because of the high level of noise, especially for low-bit quantization. Thus, we used a QAT algorithm to mitigate the quantization noise. This algorithm needs a pre-trained LCDPDNN in order to provide satisfying results. Next, it works by quantizing the NNs when producing its predictions, and performs a GD using floating-point numbers to avoid vanishing gradients. Doing so, the LCDPDNN learns how to correct the quantization noise. Results of numerical simulations show that using QAT, we can enable an efficient low-bit DPD perfectly linearizing the considered PA. Furthermore, we proposed an FPGA configuration of the LCDPDNN for further implementation. Through a synthesis, we showed that the resource consumption is significantly reduced compared to a 32-bits DPD. This means that we can expect a lower energy consumption and a diminished latency.

Nonetheless, the current implementation of the LCDPDNN on FPGA is still static, and do not take into account the time-varying aspects described in the previous chapter. Then, the adaptive algorithms developed in Chapter Section 4 shall be also implemented. For instance, this aspect is still under investigation, but one solution could be to run the MAML algorithm on an embedded processor close to the FPGA, e.g. ARM processor. That way, we can copy the updated weights on the FPGA faster.

The technical contributions of this chapter have been approved by the following conference paper.

- [C2] A. Falempin, J. Laurent, J-B. Doré, R. Zayani and E. Calvanese Strinati, “On the Performance of Quantized Neural Networks based Digital Predistortion for PA linearization in OFDM systems,” In Proc. IEEE Conference on Vehicular Technology (VTC2022-Fall), London, UK, 2022.

Conclusion and Perspectives

Contents

6.1 Overall conclusion	73
6.2 Short-term perspectives	74
6.2.1 Towards a realistic implementation	74
6.2.2 Integrating memory effects of the PA	74
6.3 Mid-term perspectives	74
6.3.1 Joint PAPR reduction and DPD	74
6.3.2 Complex-valued Neural Networks for communications	75
6.4 Long-term perspectives	75
6.4.1 Massive MIMO DPD	75
6.4.2 Energy-efficient communications	76
6.4.3 Meta-learning for communications	76

6.1 Overall conclusion

IN this thesis, we have investigated and designed a Digital Pre-Distortion (DPD) function using Neural Networks (NNs) to linearize Power Amplifiers (PAs). The goal of our study is proposing a low-complexity solution that can be easily implemented on a hardware system, and that can also cope with time-varying effects affecting the PAs such as aging, temperature and electrical variations. Doing so, we are able to improve the power efficiency of the PA while keeping away from nonlinearities damaging the wireless transmission. NNs are known to be efficient at dealing with nonlinear problems. Notwithstanding, it is challenging to conceive low-complexity NN based solutions that are energy efficient, and easily implementable. Thus, to propose an efficient DPD module based on NNs, we must think differently from the “black-box” way of designing NNs. Indeed, for instance, the existing solutions to perform DPD based on NNs are proposing wide architectures composed of multiple trainable layers of neurons. This kind of architecture leads to high complexity which cannot be considered in systems where latency and power efficiency are crucial.

To avoid this issue, in Chapter 3, we studied novel NNs architectures to perform DPD. Specifically, we designed two NNs to correct respectively Amplitude-to-Amplitude (AM/AM) and Amplitude-to-Phase (AM/PM) distortions induced by the PA. By considering custom architectures based on the characteristics of the PA, we achieve a low-complexity design with excellent performance satisfying the Long Term Evolution (LTE) and Fifth Generation (5G) standards requirements, and also valid for beyond 5G communications. Moreover, by adopting a low-complexity of the NNs, we ensure a lower latency and a better energy efficiency of the transmitter. Despite the improvements brought in terms of complexity, latency and performance, our solution remains “static” and cannot deal with time-varying effects that may affect the PA. Hence, in Chapter 4, we investigated two approaches to improve the adaptive behavior of the developed DPD. Namely, we first use a meta-learning algorithm to improve the generalization of our DPD. By training the NNs on multiple states of the PA, we can derive an optimized set of weights to initialize the DPD. Doing so, we can perform an online calibration on whatever new state of the PA with few data and lower computing time. Thus, meta-learning allows to perform an adaptive DPD at a low-cost with good linearization performance, compared to the requirements for 5G networks. In addition, we also designed an algorithm that selects the best DPD regarding the current state of the PA. It is based on a priori knowledge, and thus requires to conduct multiples characterizations of the PA under different conditions. Based on this, we can learn multiple set of weights adapted to these characteristics. Finally, our proposed weights selector will choose the closest weights for the DPD regarding the current PA. Thanks to this algorithm, we are able to propose a fast and cost-effective DPD in time-varying scenarios. Even if DPD based on meta-learning solutions can bring real interests in time-varying PA linearization, hardware implementation of such solution is challenging because of the quantization of numbers in the NNs. High-bit quantization cannot be considered for embedded systems where latency and energy consumption are important.

In Chapter 5, we studied the quantization of our proposed NNs performing DPD. When considering a hardware implementation, one must optimize the resource usage and energy consumption. However, despite the low-complexity aspect of our DPD, it uses 32-bits quantization which consumes a lot of resources and energy. Moreover, in practical systems Digital-to-Analog Converter (DAC) and Analog-to-Digital Converter (ADC) are operated using a limited number of bits. Hence, it is crucial to lower the number of bits used by our NNs. Thus, we proposed to analyze and correct the effects of a low-bit quantization on our the NNs performing the pre-distortion. We performed a low-bit quantization operation on our NNs, aided by a new training to mitigate the noise introduced by the quantization operation. Based on numerical evaluation, we observed no performance loss when the number of bits is greater or equal than 4. In addition, we considered a future implementation of the solution on a Field-Programmable Gate Array (FPGA). Namely, we proposed designs of both NNs performing respectively the AM/AM and AM/PM pre-distortions. Then, we evaluated the resource consumption thanks to a FPGA synthesis. Then, we showed that using a low-bit quantization drastically reduce the

resource usage of the **FPGA** compared to the 32 bits case. The energy consumption is then reduced.

Finally, in the thesis, we have successfully demonstrated that we can improve the energy efficiency of wireless networks by performing **DPD**. Our proposed study highlighted that **DPD** must be carefully designed and optimized to answer the raised concerns about complexity, adaptativity and implementability. Then, our proposal leverages all these criteria. Then, we can state that the conduct research will help in the development of sustainable transmitter and more generally sustainable networks.

6.2 Short-term perspectives

6.2.1 Towards a realistic implementation

A currently ongoing work is to implement the proposed **DPD** on **FPGA**. This step is straightforward since we already have the designs, and **FPGA** synthesis. This will allow to test our proposal in a real test bench to assess its reliability with a real **PA**. Moreover, an implementation would allow us to measure the power efficiency of the system, and evaluate the benefits of using our proposed **DPD**. Still, **FPGA** are consuming a lot of power. Then, it could be envisaged to implement this on a system on chip to lower the power consumption. For instance, to benefit from a lower latency and power consumption, our solution could be implemented on high efficient digital signal processing units. As an example, we demonstrated in [84], that using high efficient **DSP** unit to implement **NNs** leads to high throughput with a very low power consumption. One shall also evaluate the global gain in terms of power consumption.

6.2.2 Integrating memory effects of the PA

In this thesis, we choose to focus on the time-varying aspect of the **PA** and the optimization of the associated **DPD** to propose a lightweight solution. However, we did not take the memory effects exhibited by the vast majority of **PAs**. Memory effects create a distortion that must be corrected to have an efficient **DPD**. In our case, a solution to this could be using neural networks structures that can process time series. Indeed, since the output of the **PA** depends on past inputs, then it shall be considered to use neural structures such as recurrent neural networks, Long-Short-Term-Memory (**LSTM**) structures or even convolutional neural networks to help with the memory effects. Eventually in [85], authors proposed to use an **LSTM** architecture to deal with memory effects. Such element could help our designed **NNs** to cope with memory effects with a moderate increase of the overall complexity. Tackling both memory and time-varying effects of the **PA** has not yet been proposed in the literature, at least not with complexity and energy constraints. Then, we believe that such study will definitely improve the designs of **DPD** modules.

6.3 Mid-term perspectives

6.3.1 Joint PAPR reduction and DPD

DPD can successfully correct the nonlinearities of a **PA**. However, the high Peak-to-Average Power Ratio (**PAPR**) exhibited by Orthogonal Frequency Division Multiplexing (**OFDM**) waveform will prevent from using the **PA** at its maximum efficiency. Indeed, a high **PAPR** will cause saturation if the back off is small. Then, it must be reduced in order to use the **PA** efficiently. Thus, **PAPR** reduction techniques must be used alongside **DPD** to maximize power efficiency. More specifically, a joint optimization of both aspects must be considered because the latter exhibit mutual effects regarding the trade-off between efficiency and linearity. In [86], authors have proposed a method to jointly optimize **PAPR** and **DPD**. Despite the good results of this approach, the complexity of their solutions is challenging. To lower the complexity of this algorithm, we can consider using end-to-end learning via autoencoders. The latter are specific deep learning architectures that are training using the input as the output. It allows

optimizing both transmission and reception chains. Then, this can be used to jointly optimize the PAPR and the DPD. Actually, this has recently been studied in [87]. However, the proposed solution still exhibit high complexity and do not take time-varying effects into account. Then a future work could be an end-to-end optimization using an autoencoder that exhibit low-complexity. An interesting feature would be adding the time-varying aspects with meta-learning for example. Note that a constraint on the ACLR should be taken into account while optimizing the DPD and PAPR, respecting the 5G standard spectrum mask.

6.3.2 Complex-valued Neural Networks for communications

In wireless communications, signals are usually represented in the complex domain. However, conventional NNs architectures are performing operations on real numbers. This implies that to deal with complex numbers, one must separate real and imaginary parts of the complex number. When working on real numbers, we loose the phase information of the complex number which results in a less efficient method than one using directly complex numbers. For instance, considering the DPD, NNs using real value representation of the complex number leads to lessen performance. This is mainly due to a loss of the phase information. Thus, one can benefit from Complex-Valued Neural Networks (CVNNs) to improve the performance of such technique. More generally, in every communication problem, CVNNs can be used to bring more generalization in NNs models [88]. However, CVNNs are very challenging because of the gradient computation which is complicated while dealing with complex numbers. Moreover, most deep learning libraries, such as TensorFlow [54] and PyTorch [89] are mostly tailored to work with real numbers. Then, the research community is a bit less active on that specific field. Nonetheless, first concerning DPD, instead of proposing FCN architectures based on the Cartesian representation of the signal, one could use a CVNN to avoid the performance loss. It could also be better to envisage such NN implementation to avoid splitting the DPD into amplitude and phase correction, since the CVNN would perform a joint optimization of the amplitude and phase of the input/output signals of the PA. Differently, CVNNs might be also interesting for constellation shaping thanks to their better generalization on problems where phase information is crucial.

6.4 Long-term perspectives

6.4.1 Massive MIMO DPD

In this thesis, we focused on Single-Input Single-Output (SISO) transmission, meaning that we only consider one antenna unit. In order to enhance the spectral and energy efficiencies, the 5G considers adopting the massive MIMO technology due to its ability to provide good quality of service and to support numerous users demanding different services. When using massive MIMO, classically, we employ multiple Radio-Frequency (RF) chains with one PA per transmission chain. This means that one would need to correct the nonlinearities of all the PAs using DPD. A naive approach could be replicating our low-complexity DPD to each RF transmission chain. This could result in a high complex system due to the high number of DPDs needed. Eventually, in [90], authors proposed precoding techniques to avoid such complexity. When considering hybrid beamforming architecture, a phase shifter is used before the PA. In that case, performing DPD is challenging because it has to take this aspect into account. Besides, some recent work has emerged to propose performing DPD before the precoding operation. This allows having a unique DPD instead of one for each antenna, thus lowering the system complexity. Still, there is a need to consider the time-varying aspect of the PA which is challenging in this proposal. One way to counteract this problem would be using meta-learning explored in this thesis. Namely, a meta-learning would be conducted using all PAs of the MIMO system. Then, we would end up with a generic DPD that can be used for each PA of the MIMO array. Then, a fine-tuning step can be used to refine each DPD.

6.4.2 Energy-efficient communications

In this thesis, we mainly focused on the DPD for PA since the latter is the most power-hungry component of the network. By performing DPD, we greatly improved the power-efficiency of the PA. However, both RF transmission and reception chains must be optimized for optimum energy-efficiency. Hence, a potential lead would be to add the energy consumption as a metric of the global transmission/reception chain and optimize it. The main challenge of this is to derive an energy consumption model of the network since it is highly dependent on the underlying hardware. For instance, in [91], authors considered a joint optimization of energy consumption, precision and performance in federated learning scenario. Nevertheless, this study considers a specific chip tailored to execute Convolutional Neural Networks for image processing. Thus, it is hardly reproducible for any NN architecture. Then, it shall be interesting to consider a joint optimization of the energy consumption, EVM and ACLR to have the most efficient system. The most difficult part is finding an energy consumption model. This could be achieved by doing measures on some different hardware, e.g. FPGA, Graphics Processing Units (GPUs) boards [92], and derive an energy consumption model.

6.4.3 Meta-learning for communications

In Chapter 4 of this thesis, we studied the use of meta-learning for DPD. Despite the satisfying results, we are still investigating to improve its performance. Besides, this class of algorithms could be used more widely in communications. Actually, meta-learning is made to improve the generalization of NNs, and is then pertinent when the problem to solve is mainly time-varying. In our investigation, we used it for DPD, but it could also be used in channel estimation because the channel highly depends on its environment. For instance, the MAML algorithm could be used to train a NN on a time-varying channel considering numerous tasks for which we consider one state of the channel. Then, during online usage, the NN would use fewer pilots than a full channel estimation. More generally, meta-learning can be used to add some intelligence to every system that needs periodic updates and/or generalization.

Appendices

Résumé étendu de thèse

DANS ce manuscrit de thèse, nous présentons des travaux qui s'articulent autour de l'amélioration de la consommation énergétique des réseaux de télécommunications sans-fil. Plus particulièrement, nous nous intéressons à l'efficacité énergétique des transmetteurs radio-fréquence (RF). Les amplificateurs de puissance présents dans ces chaînes constituent les principaux éléments énergivores qu'il faut optimiser pour réduire la consommation énergétique. Ainsi, nous proposons la conception d'une fonction de pré-distorsion numérique basse consommation, basée sur des techniques d'apprentissage profond, pour améliorer l'efficacité de l'amplificateur. Tout d'abord, nous introduisons des éléments contextuels afin de comprendre pourquoi une telle étude s'avère indispensable. Ensuite, nous résumons brièvement les travaux conduits dans cette thèse.

A.1 Chapitre 1 : Introduction

La société mondiale connaît actuellement une numérisation de tous les services, en particulier liés à l'utilisation d'équipements mobiles. Au fil des décennies, les technologies de communications sans-fil se sont largement imposées auprès des utilisateurs. On constate notamment que ces dernières sont en perpétuelle évolution dans le but de répondre nouveaux besoins des utilisateurs. Ainsi, tout commença au début des années 80 avec l'apparition de la première génération de communications sans-fil permettant de passer des appels. Il s'ensuivit dans les années 90, l'apparition de la deuxième génération signant le début des communications numériques avec l'envoi de messages et l'accès aux services de données. Un peu plus tard, la 3G [2] apparut avec une amélioration du service de données. Ce n'est qu'en 2009 que la 4G [3] s'installe améliorant grandement les débits.

Cependant, nous constatons l'émergence de nouveaux services tels que la conduite autonome, la réalité virtuelle, la télémédecine ou bien encore l'Internet des objets qui font l'objet d'une demande accrue en bande passante et d'un accès quasi-instantané aux réseaux. Récemment, la 5G, en cours de déploiement, introduit des changements majeurs dans la vision des réseaux de communication. Les nouvelles architectures du cœur de réseau et interfaces radios permettent de répondre à de nouveaux marchés verticaux. Ainsi, la 5G se construit autour de trois scénarios [5] :

- **eMBB**: cela adresse les applications très haut débit, jusqu'à 20Gbit/s en débit descendant [6]. Ce scénario met en exergue une couverture réseau plus large pour fournir un accès continu dans les zones denses;
- **URLLC**: ce cas d'usage introduit une disruption dans les technologies de communications. Cela concerne de nouvelles applications ayant besoin d'une très faible latence. Cela concerne, entre autres, la chirurgie à distance, l'industrie 4.0 et les réseaux électriques intelligents.
- **mMTC**: ici, on s'intéresse plutôt à la connexion de milliards d'objets avec une consommation de puissance et un usage de données réduits.

A.1.1 La boîte à outils du champion

Pour répondre aux exigences mentionnées ci-dessus, de nouvelles technologies font leur entrée. Pour atteindre de hauts débits, la 5G tire parti des bandes millimétriques qui utilise de hautes fréquences (>

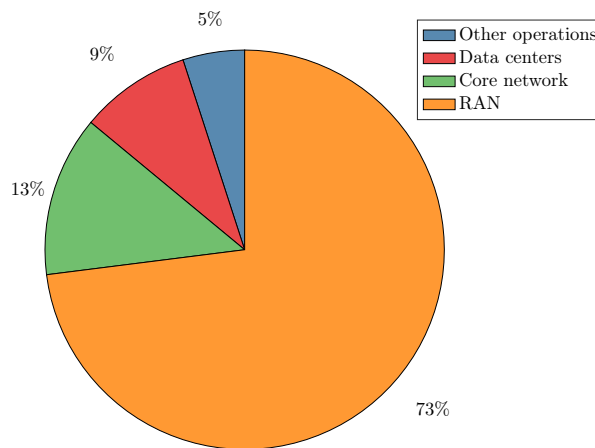


Figure A.1: Consommation énergétique des opérateurs

24GHz). Cependant, les signaux hautes fréquences souffrent de problèmes d'affaiblissement, atténuation, blocage et directivité dus à une petite longueur d'onde. Les systèmes à très grand nombre d'entrées multiples et à sorties multiples ("massive MIMO" en anglais) permettent d'alléger le problème mis en évidence par les bandes millimétriques [7]. Par ailleurs, le choix des formes d'ondes radios ont un fort impact sur la conception des transmetteurs et récepteurs (RF). À l'heure actuelle, le multiplexage par répartition orthogonale de la fréquence (OFDM) est considéré pour la 4G et la 5G [8]. En effet, cette forme d'onde possède de nombreux avantages, résistante à la sélectivité en fréquences et aux interférences inter-symboles. Nonobstant, cette technologie présente un inconvénient majeur à savoir un haut rapport entre puissance maximale et puissance moyenne ("PAPR" en anglais). Cela pose un défi majeur pour les conceptions d'amplificateurs de puissance. En dépit de l'attractivité pour ces techniques, il n'en reste pas moins qu'un défi de taille est encore à relever, l'efficacité énergétique des réseaux.

A.1.2 Le défi du siècle : Des réseaux durables

Bien que l'amélioration des performances ne cesse de se poursuivre, il devient nécessaire et critique de s'intéresser à l'efficacité énergétique des réseaux. Avant l'arrivée de la 5G, la consommation de puissance a souvent été négligée au profit de la performance. La 5G commence à considérer l'aspect efficacité énergétique en utilisant de nouvelles technologies [10] telles que le "massive MIMO", veille intelligente et apprentissage automatique. Malgré ces avancées, en l'état, il reste toujours des améliorations à apporter côté efficacité énergétique. À priori, il semblerait que les feuilles de route des futures technologies de communications, telles que la 6G, prennent activement en compte cet aspect [11]. Ainsi, la question en or demeure : *Comment améliorer l'efficacité énergétique des réseaux de télécommunications ?* Répondre à cette question requiert une analyse globale de la consommation de puissance des systèmes de communications sans-fil.

La Figure A.1 présente la consommation énergétique moyenne des opérateurs réseaux [12]. On note que la majorité de la consommation énergétique provient du RAN. La station de base est donc l'élément le plus consommateur en énergie. Spécifiquement, nous pouvons souligner grâce aux Figure A.2a et Figure A.2b que les optimisations énergétiques sont à prévoir sur le traitement radio opéré par la station de base.

Plus précisément, nous pouvons voir que l'amplificateur de puissance (PA) consomme 59% de la puissance allouée à une unité d'antenne. Par conséquent, il est important d'optimiser son efficacité. Cependant un PA présente des non-linéarités qui doivent être évitées pour assurer la qualité de la transmission.

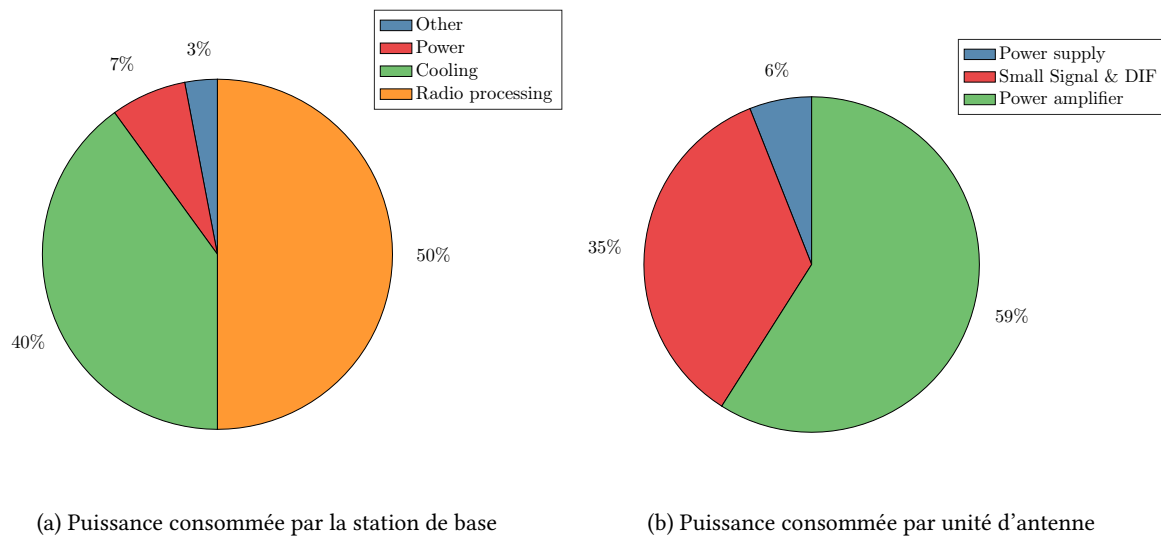


Figure A.2: Puissance consommée par le traitement radio

A.1.3 Les amplificateurs de puissance

Un PA est un composant matériel qui convertit un signal de faible intensité en un signal de haute intensité. Idéalement un PA doit appliquer un gain à son entrée résultant en une sortie amplifiée linéairement. Cependant, un PA est non linéaire dû à des transistors qui présentent des régimes non linéaires. Ainsi, un PA amplifiera un signal son signal d'entrée, mais des distorsions seront injectées ce qui causera des erreurs de détection au niveau du récepteur.

Afin de comprendre pourquoi des activités de recherche sont requises pour améliorer l'efficacité du PA, nous présentons quelques éléments sur l'impact de l'utilisation conjointe de l'OFDM et le PA dans la chaîne RF.

Non-linéarités Un PA présente des non-linéarités qui peuvent être plus ou moins sévères en fonction de son régime de travail, qui dépend de l'amplitude du signal d'entrée. En l'état, lorsqu'un signal OFDM existe le PA dans sa zone non linéaire, ce dernier va induire un haut niveau de distorsion à sa sortie. Cela implique donc qu'il sera impossible de retrouver l'information émise à partir du signal reçu. Il convient donc de corriger ces non-linéarités.

Par ailleurs, les signaux OFDM présentent un PAPR élevé, autrement dit le signal est sujet à d'importantes fluctuations en puissance. Cela peut donc faire entrer le PA dans un régime de saturation, induisant d'importantes perturbations sur le signal transmis.

Efficacité énergétique L'efficacité énergétique d'un PA est maximale proche de sa saturation. Cependant, les non-linéarités sont plus fortes dans cette zone. Comme expliqué précédemment, le haut PAPR de l'OFDM causera la saturation du PA. Pour éviter cela, on considère souvent d'abaisser la puissance moyenne du signal [14]. Appelée "back-off", cette opération permet de travailler dans le régime linéaire du PA. En revanche, l'efficacité énergétique du composant descend à son minimum.

Donc, il convient d'optimiser conjointement la consommation énergétique du PA ainsi que ses non-linéarités, *i.e.* linéariser le PA, tout en maximisant son efficacité énergétique.

A.1.3.1 Optimiser l'efficacité du PA

Deux solutions principales existent pour exploiter au mieux le PA: i) la réduction du PAPR des formes d'ondes, et ii) la linéarisation du PA.

Réduction du PAPR Réduire le PAPR des formes d'onde permet de minimiser le "back-off" introduit. En effet, moins de fluctuations de puissance impliquent que l'on peut opérer le PA plus proche de sa saturation. Plusieurs études ont été conduites dans le but d'améliorer le PAPR des systèmes OFDM [15].

Par exemple, il est aussi possible d'optimiser la constellation transmise [16] [17] afin de réduire le PAPR. En outre, certaines études proposent des nouvelles formes d'ondes avec un PAPR réduit [18] [19].

Ainsi, ces techniques permettent d'améliorer le PAPR des signaux. Cependant, même en évitant la saturation du PA, il y aura toujours des non-linéarités.

Linéarisation du PA Pour corriger les non-linéarités du PA, il existe plusieurs méthodes [20]. On dénote trois grandes techniques : "feedforward", "feedback" et pré-distorsion.

La pré-distorsion numérique (DPD) reste la technique la plus en vogue à l'heure actuelle. Elle consiste en l'ajout d'un module avant le PA de telle sorte que le système résultant, DPD et PA, soit linéaire. Bien que largement employé, les recherches sont toujours d'actualité pour améliorer la complexité et l'efficacité énergétique. De plus, le PA est sujet à des effets variables dans le temps tels que la température et les variations électriques. Ainsi, idéalement la pré-distorsion devra prendre en compte ces effets.

A.1.4 Apprentissage automatique pour les télécommunications

Précédemment, nous avons vu que l'évolution vers de nouvelles technologies implique de nouveaux défis à résoudre. Ces derniers sont souvent des problèmes non linéaires et non convexes, pour lesquels on ne trouve pas de solution évidente avec les outils d'analyse classiques. Ainsi, les techniques d'apprentissage automatique ont émergé pour résoudre ce type de problème. Brièvement, l'apprentissage automatique regroupe une catégorie d'algorithmes capables d'apprendre à résoudre automatiquement un problème à partir de données le caractérisant. En particulier, l'apprentissage profond a fait ses preuves dans la correction des non-linéarités. En l'occurrence, ces techniques peuvent être utilisées pour résoudre des problèmes liés à la couche physique des télécommunications [24] [25].

Par ailleurs, il a également été envisagé d'optimiser l'efficacité du PA avec des réseaux de neurones (NNs), notamment par réduction du PAPR [26] ou pré-distorsion numérique [27]. Cependant, les solutions basées sur l'apprentissage profond mettent en avant une complexité élevée et donc une réalisation pratique difficile. Il devient donc évident de s'intéresser à ces aspects étant donné que les communications durables et à faible coût sont envisagées dans le futur. Ainsi, les travaux présentés dans cette thèse portent sur la linéarisation de PA en utilisant une DPD basée sur des NNs, en prenant en compte les contraintes mentionnées auparavant.

A.2 Chapitre 2 : La pré-distorsion numérique pour amplificateurs de puissance

Comme énoncé, les PAs présentent des comportements non linéaires qu'il convient de corriger pour maximiser le compromis entre linéarité et efficacité énergétique. Pour ce faire, nous considérons l'emploi d'une pré-distorsion numérique largement utilisée aujourd'hui [28]. Pour mieux comprendre l'impact du PA dans les systèmes de communications, nous présentons quelques éléments importants sur la modélisation de ce dernier et comment fonctionne la DPD.

A.2.1 Modélisation du PA

Il existe plusieurs manières de modéliser un PA en fonction de l'aspect que l'on souhaite étudier. Il existe trois grandes catégories de modèles : modèles sans mémoire, modèles avec effets mémoire et modèles variables dans le temps.

Modèles sans mémoire: Les modèles sans mémoire permettent de modéliser des PAs induisant des distorsions sur l'amplitude et la phase du signal d'entrée, sans dépendance par rapport à la fréquence de ce dernier. On peut notamment citer les modèles de Rapp [29] et Saleh [31].

Modèles avec effets mémoire: Lorsque la caractéristique du PA dépend de la fréquence du signal d'entrée, ce dernier présente des effets "mémoire". Autrement dit, la sortie dépend à la fois des entrées

courantes et passées. Ce type de phénomène peut s'observer sur des signaux larges bandes.

Pour modéliser ce comportement, il est possible d'utiliser des séries de Volterra [32] ou bien des modèles alternatifs à complexité réduite, tels que les modèles de Wiener et Hammerstein [33] ou bien encore modèles polynomiaux [34].

Modèles variables dans le temps: Un PA peut être sujet à des effets variables dans le temps tels que le vieillissement, la température ou bien encore les variations électriques. Ce type d'effet est souvent négligé dans la littérature. Tout de même, dans [35], les auteurs considèrent une variation des paramètres définissant le modèle de Saleh.

Dans nos travaux, nous considérons un modèle de Rapp variable. L'équation (??) présente les distorsions AM/AM et AM/PM induites par le modèle.

$$f_\rho(u) = \frac{Gu}{\left(1 + \left|\frac{Gu}{V_{sat}}\right|^{2p}\right)^{\frac{1}{2p}}}, \quad (A.1)$$

$$f_\Phi(u) = \frac{Au^q}{\left(1 + \left(\frac{u}{B}\right)^q\right)},$$

où u est l'amplitude du signal d'entrée. G représente le gain dans la région linéaire et V_{sat} la tension de saturation. p contrôle la linéarité du modèle. De plus, A , B et q sont des paramètres fixés par le 3GPP [30] pour modéliser un PA travaillant avec des fréquences supérieures à 6GHz. Pour modéliser la variation de ce modèle, nous considérons la variation du paramètre p avec $0.5 \leq p \leq 1.5$.

A.2.1.1 Compromis entre efficacité énergétique et linéarité

La Figure A.3 présente la caractéristique AM/AM d'un PA avec son efficacité (PAE). On peut remarquer un PAPR élevé imposant un recul du signal pour éviter la saturation du PA. Cela implique donc une efficacité réduite avec des non-linéarités toujours présentes. Ainsi, il faut corriger ces non-linéarités pour espérer avoir une meilleure efficacité du PA.

A.2.2 Pré-distorsion numérique : Linéariser un PA

La DPD est une technique de linéarisation qui consiste à ajouter un module avant le PA, de telle sorte que le système (DPD+PA) soit linéaire.

A.2.2.1 Modélisation

Pour modéliser une DPD, on veut s'approcher le problème suivant :

$$H_{PA} \circ H_{PD} = I, \quad (A.2)$$

où H_{PD} est la fonction de transfert de la DPD, et H_{PA} la fonction de transfert du PA. I représente la fonction identité.

Pour réaliser cette DPD, il existe plusieurs approches dépendant du PA considéré. On recense les tables de correspondance (LUTs) [38], les séries de Volterra [40] et l'estimation polynomiale [41]. Cependant, ces méthodes présentent une complexité assez élevée et ne sont donc pas adaptées pour des scénarios temps réel.

Ainsi, il est possible d'employer des techniques d'apprentissage profond, réseaux de neurones, pour modéliser la DPD. Cela a déjà été considéré auparavant dans [43]. Nonobstant, la complexité de ces techniques reste toujours élevée à cause du paradigme de la "boîte noire". Nos travaux se concentrent donc sur la proposition d'une DPD faible complexité, basée sur des réseaux de neurones, et capable de s'adapter à des effets variables dans le temps.

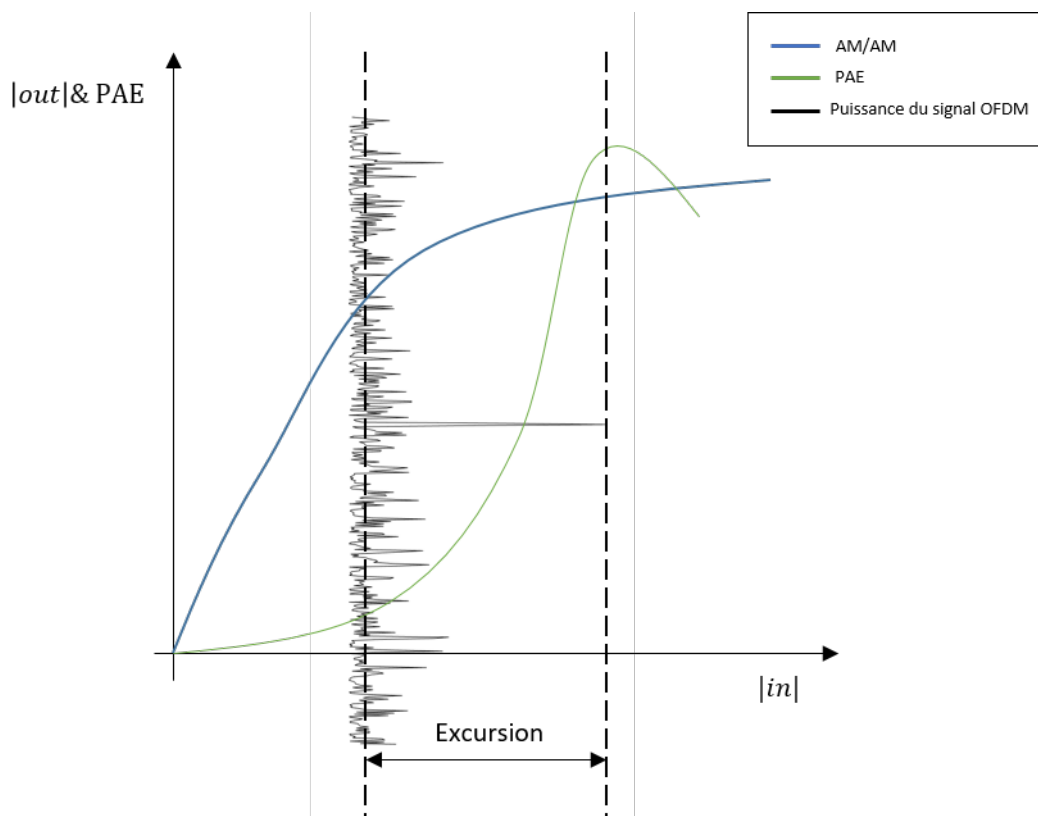


Figure A.3: Impact d'une excursion de signal élevée sur la PAE du PA

A.3 Chapitre 3 : DPD basse complexité à base de réseaux de neurones (LCDPDNN)

Dans la littérature, il a largement été prouvé que les réseaux de neurones s'avèrent très efficace pour modéliser des DPD. Cependant, la majorité des solutions tels que [43], [44], [45], présentent une complexité importante qui requiert donc d'importantes ressources de calcul. Par ailleurs, la latence de ces solutions les rend peu adaptées à des usages temps réel.

Pour améliorer la complexité de la fonction de pré-distorsion, il est nécessaire de considérer plusieurs aspects. D'une part, la décomposition du signal dans le domaine polaire permet de corriger chaque distorsion séparément. Cela permet d'avoir de meilleures performances que l'utilisation de signaux dans le domaine cartésien. D'autre part, pour chacune des distorsions, nous proposons une architecture de réseau de neurones totalement dédiée et spécifique au problème à résoudre. Ce faisant, il est possible de réduire drastiquement la complexité de la DPD.

A.3.1 Architecture des réseaux de neurones pour la DPD

La DPD proposée est composée de deux réseaux de neurones dont chacun corrige respectivement les distorsions AM/AM et AM/PM. L'architecture de ces NNs est présentée sur la figure Figure A.4. On constate que cette architecture est totalement personnalisée pour corriger chacune des distorsions du PA. Cela permet d'atteindre une conception basse complexité contrairement aux solutions reposant sur la conception à base de "boîtes noires".

Apprentissage L'apprentissage de ces réseaux utilise d'une part le signal d'entrée du PA et d'autre part le signal reçu après passage dans le canal de transmission, faisant office de chaîne de retour. Le réseau corrigeant la distorsion AM/AM a besoin de l'amplitude d'entrée du PA et l'amplitude du signal reçu pour exécuter sa phase d'entraînement. Similairement, le réseau corrigeant la phase a aussi besoin

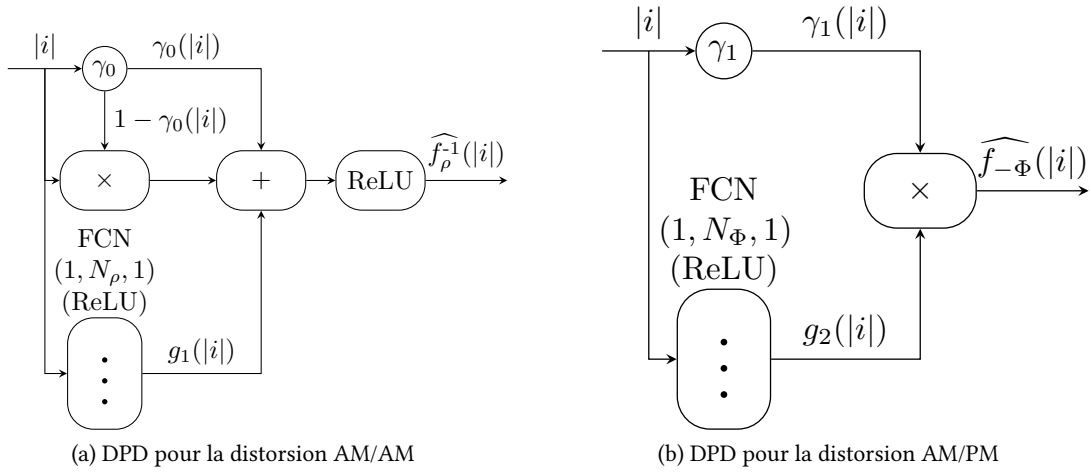


Figure A.4: Architecture du LCDPDNN pour les distorsions AM/AM et AM/PM

Table A.1: Comparaison d'algorithmes de DPD

Travaux connexes	ACLR Gain	EVM	FLOPs	NFLOPs
RT2DNN [46]	12dB	N/S	412	1
DPD w/o ILA [45]	N/S	N/S	184	0.45
Cartesian Dense DPD [43]	9dB	-32dB	322	0.78
Polar Dense DPD [44]	11dB	-37dB	162	0.39
LCDPDNN	14dB	-38dB	142	0.34

de l'amplitude d'entrée du PA, et du déphasage entre le signal reçu et le signal d'entrée du PA.

La phase d'entraînement est réalisée grâce à l'optimisation d'une fonction de coût par rapport aux données de chaque réseau. On cherche à minimiser l'erreur quadratique moyenne entre la prédiction de chaque réseau de neurones et la donnée d'apprentissage. Cela se fait par le biais d'un algorithme de descente de gradient. Cette étape permet de calibrer les paramètres de chaque réseau pour les rendre aptes à corriger les distorsions du PA.

A.3.2 Analyse de performances

L'évaluation de notre solution est conduite sur un modèle de Rapp stationnaire présentant de fortes non-linéarités. Nous démontrons que notre solution présente le meilleur compromis entre performance et complexité. Voir Table A.1 pour une synthèse sur chaque métrique.

On constate que notre solution est meilleure en tout point par rapport à ce qui est proposé dans la littérature. Les métriques d'EVM et ACLR permettent de mesurer respectivement les distorsions dans la bande et hors bande. Quant aux FLOPs, ils mesurent la complexité en termes d'opérations flottantes.

Bien que notre proposition présente une faible complexité avec d'excellentes performances, elle reste toujours inadaptée à un scénario adaptatif où la caractéristique du PA change.

A.3.3 Contributions

Ces travaux ont donné lieu aux contributions suivantes :

- [C1] E. Calvanese Strinati, D. Belot, **A. Falempin** and J-B. Doré, "Toward 6G: From New Hardware Design to Wireless Semantic and Goal-Oriented Communication Paradigms," In Proc. IEEE European Solid State Circuits Conference (ESSCIRC), Grenoble, FR, pages 275-282., 2021.
- [C2] **A. Falempin**, J-B. Doré, R. Zayani and E. Calvanese Strinati, "Low-Complexity Adaptive Digital Pre-Distortion with Meta-Learning based Neural Networks," In Proc. IEEE Consumer Communications & Networking Conference (CCNC), NV, USA, pages 453-457., Feb 2022.
- [J1] **A. Falempin**, J-B. Doré, R. Zayani and E. Calvanese Strinati, "Low-Complexity Adaptive DPD: From online optimization to meta-learning," IEEE Transactions on Broadcasting, pp:pp, 2022.

A.4 Chapitre 4 : Apprendre à apprendre la DPD

Un PA présente de nombreuses non-linéarités qui viennent perturber l'amplitude et la phase des signaux. La littérature regorge des solutions pour corriger ces non-linéarités. Cependant, le PA n'est pas un composant stationnaire, *i.e.* sa caractéristique peut varier dans le temps. En effet, des effets tels que le vieillissement, la température et les variations électriques vont influencer sur le comportement du PA. La plupart des solutions de la littérature ne parviennent pas à traiter ce comportement de manière efficace. Il est tout de même possible de lancer une nouvelle phase d'entraînement mais cela requiert un grand nombre de données ce qui n'est pas envisageable pour une mise à jour en temps réel.

Pour répondre à cela, nous proposons tout d'abord l'emploi d'une méthode de "méta-apprentissage". Cette dernière représente une nouvelle classe d'algorithmes permettant d'améliorer la généralisation d'un modèle de réseau de neurones. Elle permet notamment de réduire le nombre de données nécessaires à l'adaptation ainsi que le temps de calcul pour faire converger l'apprentissage. Ce procédé est utilisé dans [62] et [63] pour réaliser un système bout en bout pour la démodulation avec peu de pilotes. Dans notre cas, nous démontrons que l'utilisation du "méta-apprentissage", et plus particulièrement l'algorithme MAML [64] est bénéfique pour l'adaptation de la DPD.

D'autre part, nous proposons également un algorithme de sélection de poids optimaux pour la DPD. Cet algorithme repose sur la connaissance *a priori* de plusieurs caractéristiques de PA. L'idée est de réaliser des tests d'hypothèses entre ces caractéristiques et la caractéristique courante du PA pour en déduire laquelle est la plus proche.

A.4.1 Méta-apprentissage: MAML

Le méta-apprentissage est un nouveau concept d'apprentissage qui consiste à "apprendre à apprendre". L'idée est d'améliorer un algorithme d'apprentissage par le biais de plusieurs entraînements [66].

Pour réaliser du méta-apprentissage sur le PA, il est tout d'abord nécessaire de générer des tâches. Chaque tâche correspond à une caractérisation du PA. Ici, le PA est modélisé par le modèle de Rapp variable dans le temps où l'on considère la variation du paramètre p . Une tâche correspond donc à une valeur de ce paramètre. La Figure A.5 présente l'ensemble de tâches utilisées pour le MAML.

Grâce à ces tâches, l'algorithme de MAML est à même de trouver des poids spécifiques à chaque tâche, puis d'en estimer des poids dit "généraux" qui serviront d'initialisation pour une future adaptation.

Concernant l'adaptation, nous avons développé un algorithme de sélection de données pertinentes de façon à conduire la calibration en ligne avec le moins de données.

A.4.2 Sélection de poids (DPDNNWS)

Alternativement, nous proposons une solution permettant de sélectionner efficacement des poids pour la DPD afin de s'adapter à un nouvel état du PA.

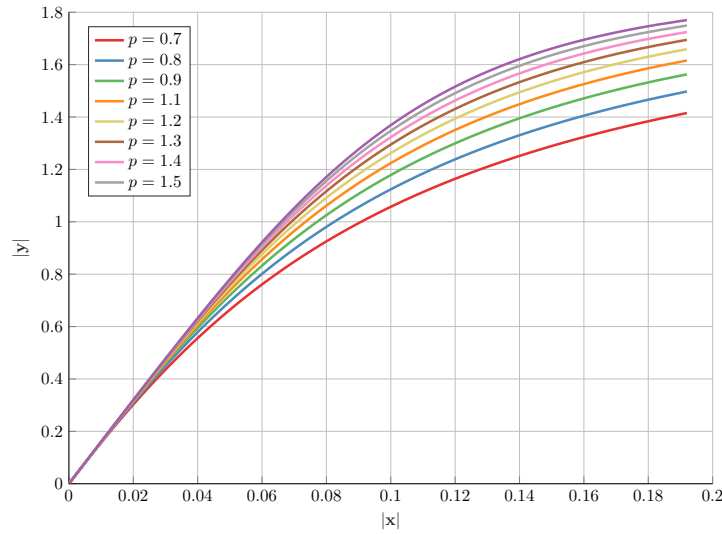
Figure A.5: Ensemble de tâches de PA $p \in [0.7, 1.5]$

Table A.2: Comparaison avec l'état de l'art

Travaux connexes	ACLR Gain	EVM	FLOPs	En ligne	Adaptabilité (Symboles IQ)
RT2DNN [46]	12dB	N/S	412	Non	Pauvre ($\approx 10^6$)
DPD sans ILA [45]	N/S	N/S	184	Non	Pauvre ($\approx 10^4$)
Cartesian Dense DPD [43]	9dB	-34dB	322	Non	Pauvre ($\approx 10^6$)
Polar Dense DPD [44]	11dB	-34dB	160	Non	Pauvre ($\approx 10^6$)
Our work					
LCDPDNN (CL)	14dB	-38dB	142	Non	Pauvre ($\approx 3 \times 10^5$)
LCDPDNN (MAML)	8dB	-29dB	142	Oui	Bon ($\approx 3 \times 10^3$)
LCDPDNN + DPDNNWS	14dB	-37dB	142 + 72	Yes	Excellent ($\approx 10^3$)

L'algorithme développé repose sur deux étapes majeures : i) connaissance *a priori* de plusieurs caractéristiques de PA et ii) à partir de ces caractéristiques, réalisation d'un test généralisé de rapports de vraisemblance (GRLT) [70].

Ainsi, par la réalisation de tests d'hypothèses, il est possible de trouver une règle permettant d'obtenir l'indice de la caractéristique la plus proche de l'état courant du PA. Cet indice donne donc l'ensemble de poids le plus optimal à utiliser pour la DPD.

A.4.3 Analyse de performances

Nous évaluons nos approches en utilisant un modèle de Rapp variable avec $p \in [0.7, 1.5]$. Nous les comparons à des solutions de l'état de l'art via plusieurs critères présentés dans la Table A.2.

Par ces résultats, nous montrons que nos travaux apportent un réel bénéfice lors de la correction des effets variables dans le temps, affectant les PAs. Par ailleurs, nos solutions consomment peu de données et possèdent une latence plus faible que leurs concurrents.

Malgré les améliorations apportées à la DPD en matière de complexité et adaptabilité, il n'en reste pas moins que son implémentation relève encore du défi. En effet, les solutions basées sur des NNs

mettent en avant des opérations coûteuses en temps de calcul et énergie. Cela est dû notamment à la quantification des nombres qui par défaut s'élève à 32 bits. Pour des systèmes matériels efficaces, cela n'est pas envisageable. On doit donc s'intéresser à cet aspect pour envisager une implémentation des travaux développés.

A.4.4 Contributions

Ces travaux ont donné lieu aux contributions suivantes :

- [P1] **A. Falempin**, R. Zayani and J-B. Doré, "Device for linearizing a power amplifier of a communication system by digital predistortion," Issued in July 05, 2022, US2107230.
- [J1] **A. Falempin**, J-B. Doré, R. Zayani and E. Calvanese Strinati, "Low-Complexity Adaptive DPD: From online optimization to meta-learning," IEEE Transactions on Broadcasting, pp:pp, 2022.

A.5 Chapitre 5 : Quantification de la DPD pour une implémentation matérielle efficace

Les NNs souffrent souvent d'un usage intensif en ressources de calcul qui mène à des opérations avec une latence importante, et également une consommation énergétique accrue à cause d'un schéma de quantification des nombres très précis, *i.e.* 32 bits.

Pour améliorer l'efficacité énergétique et réduire davantage la complexité des chaînes de transmissions radio, il est nécessaire de réduire le nombre de bits de notre DPD. Ainsi, en utilisant une quantification à faible nombre de bits, nous sommes en mesure d'assurer une solution efficace énergétiquement. Cependant, une telle quantification introduit un bruit qui peut détériorer significativement les performances globales du système. Pour éviter ce phénomène, il convient tout d'abord de le quantifier puis de le corriger. Cela permettra également d'envisager une implémentation matérielle sur FPGA par exemple.

A.5.1 Représentation d'un nombre

Il existe essentiellement deux manières de représenter un nombre fractionnel, l'arithmétique à virgule flottante et l'arithmétique à virgule fixe.

La représentation à virgule flottante la plus utilisée est celle du standard *IEEE-754* [72]. Elle considère qu'un nombre est codé par une mantisse et un exposant. La mantisse correspond à la concaténation des parties entière et fractionnelle du nombre, en omettant le bit de poids fort. Quant à l'exposant, il définit la position de la virgule.

Différemment, la représentation à virgule fixe considère une position de virgule identique pour tous les nombres dans l'intervalle du schéma de quantification choisi. Cette représentation permet de quantifier indépendamment les parties entière et fractionnelle du nombre considéré. Nous retenons cette représentation pour nos travaux car elle consomme moins de ressources et est plus facile à implémenter.

A.5.2 Quantification post-entraînement de la DPD (PTQ)

Pour apprécier les effets de la quantification sur la DPD, nous appliquons une quantification post-entraînement sur les poids et les sorties de chaque réseau de neurones constituant la DPD. Une telle opération va introduire un bruit important en raison d'une baisse de précision sur les nombres.

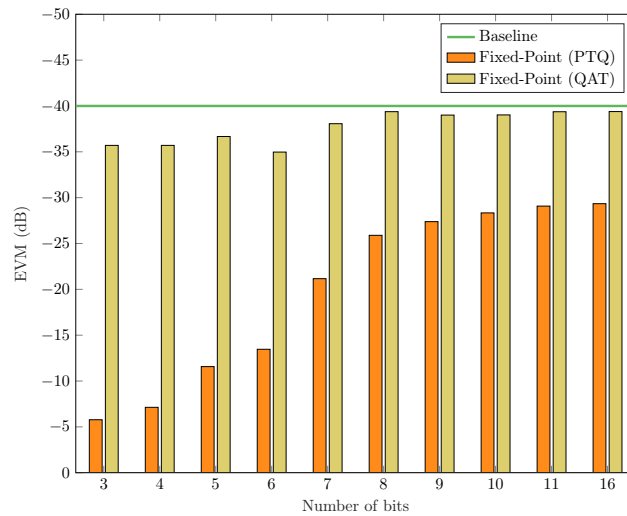


Figure A.6: Performance en utilisant PTQ et QAT

A.5.3 Apprendre à corriger l'erreur de quantification (QAT)

Pour corriger les erreurs liées à la quantification, on peut utiliser un entraînement prenant en compte ce bruit pour le corriger. Cette méthode est détaillée dans l'algorithme 2.

Algorithm 2 Entraînement avec quantification

Require: \mathcal{D} : Ensemble de données d'entraînement

Require: Taux d'apprentissage pour la descente de gradient

Ensure: Poids pré-entraînés sur 32 bits

while convergence not achieved **do**

 Quantifier le NN pendant la passe de prédiction

 Évaluer les gradients

 ► Utiliser STE pour la fonction d'arrondi

 Mettre à jour les poids

end while

Cet algorithme permet donc d'apprendre à corriger le bruit de quantification. La spécificité majeure est que la quantification n'est utilisée que dans la passe de prédiction et non lors de la rétro-propagation des gradients. Le STE [56] est un estimateur permettant d'approximer la dérivée de la fonction arrondie à 1, évitant ainsi les problèmes d'évanouissement du gradient.

A.5.4 Analyse des performances

Nous montrons que l'emploi du PTQ est très néfaste sur les performances du système. À l'inverse, le QAT permet de corriger presque en totalité les imperfections dues à l'opération de quantification. La Figure A.6 présente l'EVM du système de transmission avec PTQ et QAT. Nous observons que le PTQ dégrade fortement les performances. Le QAT met en avant des performances très proches d'un cas d'utilisation sur 32 bits.

A.5.5 Implémentation sur FPGA

Une fois la quantification effectuée, nous pouvons envisager une implémentation sur matérielle. Nous choisissons le FPGA dans un premier temps, car il est plus facile de prototyper notre DPD à base de NNs.

Nous proposons donc une conception de chaque NN pour FPGA et effectuons une synthèse de ces conceptions. La synthèse permet de générer la configuration matérielle du FPGA et ainsi d'estimer l'utilisation en ressources en fonction du nombre de bits utilisés pour la quantification. Nous montrons qu'une quantification avec un faible nombre de bits permet de réduire drastiquement la consommation en ressources du FPGA. Donc, nous optimisons l'efficacité énergétique de la DPD et par extension de toute la chaîne RF.

A.5.6 Contributions

Ces travaux ont donné lieu aux contributions suivantes :

- [C2] **A. Falempin**, J. Laurent, J-B. Doré, R. Zayani and E. Calvanese Strinati, "On the Performance of Quantized Neural Networks based Digital Predistortion for PA linearization in OFDM systems," In Proc. IEEE Conference on Vehicular Technology (VTC2022-Fall), London, UK, 2022.

A.6 Conclusion

Dans les travaux présentés par cette thèse, nous avons mis en évidence la conception d'une fonction de pré-distorsion permettant de corriger les non-linéarités du PA et donc d'améliorer son efficacité énergétique. Spécifiquement, la fonction de pré-distorsion proposée est réalisée grâce à des réseaux de neurones. L'architecture dédiée permet d'obtenir une faible complexité assurant une faible latence et une meilleure efficacité énergétique. Par ailleurs, le PA n'étant pas un composant stationnaire, il est nécessaire de compenser les effets qui peuvent affecter sa caractéristique. Particulièrement, les distorsions en amplitude et phase dépendent par exemple du vieillissement, de la température et des variations électriques du composant. Pour contrer ces effets, nous avons mis en place des approches permettant d'adapter la DPD à une variation affectant le PA. Par le biais du méta-apprentissage, nous proposons une DPD à base de réseaux de neurones très généralisable et qui peut s'adapter rapidement à une nouvelle configuration du PA. Nous avons également proposé une approche différente qui permet de trouver la DPD la plus proche de l'état actuel du PA pour obtenir la linéarisation la plus efficace. Enfin, afin d'optimiser de manière significative la consommation énergétique de notre DPD, nous avons conduit une étude sur la quantification de cette dernière afin de proposer une implémentation à faible nombre de bits. Cela permet notamment de diminuer grandement la consommation énergétique des réseaux de neurones composant la DPD. Finalement, nous pouvons affirmer que les travaux proposés dans cette thèse permettent d'offrir une DPD basse complexité, adaptative et efficiente énergétiquement.

Design and Analysis of MIMO Systems Using Energy Detectors for Sub-THz Applications

Design and Analysis of MIMO Systems using Energy Detectors for Sub-THz Applications

Simon Bicaïs, Alexis Falempin, Jean-Baptiste Doré, Valentin Savin
CEA-LETI, Université Grenoble Alpes, F-38000 Grenoble, France
jean-baptiste.dore@cea.fr

Abstract—The significant amount of unused spectrum in sub-TeraHertz frequencies is contemplated to realize high rate wireless communications for beyond 5G networks. Yet, the performance of radio-frequency sub-TeraHertz systems is severely degraded by strong oscillator phase noise. Therefore, we investigate in this paper the use of multiple-input multiple-output (MIMO) systems with energy detection receivers to achieve high rate communications robust to phase noise. First, the design of the receiver detection algorithm is addressed. Two detectors are proposed for the studied nonlinear MIMO channel, either derived from the maximum likelihood decision rule by using a Gaussian approximation, or based on the use of neural networks. Second, the communication performance is assessed through numerical simulations for uncoded and coded systems. We consider a realistic scenario modeling an indoor wireless link in D-band with directive antennas and strongly correlated line-of-sight channels. Our results demonstrate that spatial multiplexing with non-coherent sub-TeraHertz transceivers can be realized on strongly correlated line-of-sight channels using the proposed detection schemes. Thereby, we highlight that high-rate radio-frequency sub-TeraHertz systems can be implemented with low-complexity and low-power architectures using MIMO systems with energy detection receivers.

Index Terms—Sub-TeraHertz communications, Physical layer, MIMO, Envelope detectors, Detection algorithms, Neural networks.

I. INTRODUCTION

Considering the spectrum shortage in cellular bands, the interest for communications in the *TeraHertz* (THz) spectrum from 0.1 THz to 1 THz is continuously growing [2]. THz frequencies offer a significant amount of unused bands [3] and represent an opportunity to achieve high data rate wireless communications. Radio-frequency (RF) THz communication systems are envisaged to meet the requirements of beyond 5G networks. This paper focuses on the use of the sub-THz spectrum from 0.1 THz to 0.3 THz, in which bands of several tens of GHz are expected to be allocated to fixed and mobile services. In particular, we investigate one of the contemplated sub-THz applications [3]: a fixed indoor high data rate wireless link. Nonetheless, this paper provides general results relevant to other applications such as enhanced hot-spot *kiosk* or device-to-device communications.

To achieve high data rate sub-THz communications, additional research is required to design efficient and new physical layer algorithms. Traditional techniques cannot be directly transposed to sub-THz bands as they do not consider the

specific features of RF impairments in sub-THz systems. In particular, they suffer from strong phase impairments due to the poor performance of high-frequency oscillators [4]. Indeed, as the carrier frequency increases, the phase noise performance of oscillators largely deteriorates. These strong phase impairments limit the achievable data rate of sub-THz systems, notably by causing detection errors. Therefore, state-of-the-art approaches [3] [5] investigate the use of coherent systems together with channel bonding, also referred to as carrier aggregation. This type of architecture needs to be further combined with signal processing optimizations [6], to mitigate the effects of phase impairments leading to complex practical implementations. Alternatively, one may consider the algorithm in [7] to mitigate the phase noise. Specifically, this algorithm allows to estimate and whiten the interference due to the phase impairments, based on the derivation of an approximate maximum likelihood detector. However, we consider in this study non-coherent detection, for its inherent robustness to phase noise and simple implementation. In contrast to the conventional approaches using linear RF chains, our purpose is to enable high-rate sub-THz communications using low complexity transceivers, employing energy detectors and suitable transmission schemes. In this regard, it is worth mentioning that a fully integrated 260 GHz on-off keying (OOK) transceiver was demonstrated in [8]. Transceivers based on energy detection (ED) have been extensively studied for systems with a single transmit antenna and multiple receive antennas, see [9] and references therein. Nevertheless, for non-coherent sub-THz systems, the main challenge is to increase the spectral efficiency. With regard to this objective, the work in [10] is relevant as it shows that multiple-input multiple-output (MIMO) systems with amplitude detection receivers may exploit spatial multiplexing to increase their spectral efficiency. Therefore, we investigate the design of MIMO systems with ED receivers to achieve high rate sub-THz communications.

This paper extends the work in [10] by analyzing and evaluating the system performance in a sub-THz scenario. In contrast to [10], where the channel is built from independent and identically distributed (i.i.d.) Gaussian entries, we assume that each spatial stream is transmitted with a directive antenna on a line-of-sight (LoS) channel. Consequently, the channels are strongly correlated, and moreover, the resulting interference is nonlinear due to the ED at the receiver. The strong and nonlinear interference between channels represents a significant challenge to achieve spatial multiplexing gain

This paper has been submitted in part to the 2020 IEEE International Symposium on Personal, Indoor and Mobile Radio Communications [1].

with non-coherent transceivers in sub-THz frequencies. The contributions of the paper are the following. First, we derive an analytical model for MIMO systems using ED receivers in sub-THz bands. Second, the design of the receiver detection algorithm is investigated. We derive a detector corresponding to the studied nonlinear MIMO channel using a Gaussian approximation approach, which we refer to as the Maximum Likelihood Detector with Gaussian approximation (MLD-GA). In addition, we propose an original and efficient detector based on neural networks, which does not require any knowledge of the channel or assumption on it. We also detail the differences between state-of-the-art detection methods and the two proposed detectors. Third, the system performance is evaluated through numerical simulations. We introduce a realistic scenario modeling a fixed indoor wireless link in the D-band at 145 GHz. Our results show that communications can be achieved with strong spatial interference between channels using the proposed detection algorithms. Fourth, we consider the integration of a forward error correction (FEC) scheme in order to achieve channel coding gains. With regard to the targeted high-rate low-complexity applications, we propose the use of a Bose, Ray-Chaudhuri and Hocquenghem (BCH) code with a short packet length that can be implemented with a low-latency, low-complexity decoder. The results of numerical simulations confirm that the integration of the FEC scheme leads to significant performance gains in terms of achievable data rate. It should be mentioned that this work is an extension of our initial study [1] in which the system model and the detector have been introduced. In addition to [1], we conduct a more thorough analytical description. Moreover, we conduct a fine analysis of the detectors behavior through numerical simulations. We also present a practical implementation and the limitation of the mathematical system model. Notably, the practical radiation pattern of a real D-band antenna is considered and its performance is discussed and compared against the sectorized antenna model, widely considered in the literature.

The main contribution of this paper is to demonstrate that spatial multiplexing with non-coherent sub-THz transceivers can be realized on strongly correlated LoS channels. Thereafter, this paper highlights that MIMO systems with ED receivers offer a valuable solution to achieve high rate communications in sub-THz frequencies with low-complexity and low-power RF architectures.

We also discuss the opportunity to derive a detector based on deep learning for such a non linear system model. Indeed, there has been a growing interest in wireless communications regarding the use of deep learning techniques for communications as it is addressed in [11] [12] and [13]. More specifically, [13] provides a general overview of the use of neural networks for detection in communication systems and also for linear MIMO systems. In [14], the use of deep neural network detection is investigated for linear MIMO systems with the knowledge of the propagation matrices. In [15] and [16], the authors present neural network detectors both based on a linear MIMO system with channel and/or noise distribution knowledge. Nevertheless, none of these works consider the use of neural networks for nonlinear MIMO systems. Similarly to

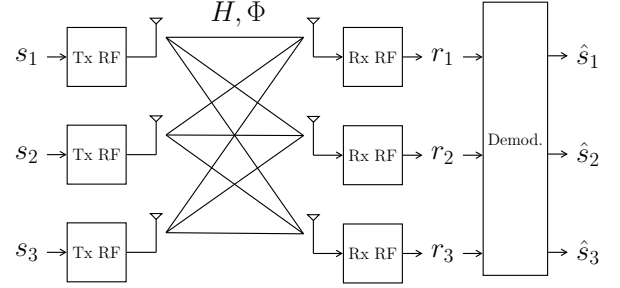


Fig. 1: Block diagram of a 3×3 MIMO transceiver

the proposed neural networks based detector, the combination of neural networks with energy detectors is addressed in [17]. However, in [17] neural networks serve a different purpose and aim to detect the unused portions of the spectrum in order to communicate in an opportunistic way, *i.e.* spectrum sensing applications. To the best of authors' knowledge, the proposed neural network based detector for nonlinear MIMO system with ED receivers is original and not addressed in the literature. Finally, we also discuss the implementation of the proposed detection schemes in practical systems.

The remainder of this paper is structured as follows. Sec. II outlines the system model. In Sec. III, the design of the receiver detection algorithm is addressed. Sec. IV is dedicated to the performance analysis. Sec. V discusses several considerations related to the practical implementation of the proposed detectors. Eventually, Sec. VI concludes the paper.

II. SYSTEM MODEL

We consider a MIMO communication system with N_t transmit antennas and N_r receive antennas with $N_t \leq N_r$, as illustrated in Fig. 1. The propagation channel is described by two $N_r \times N_t$ matrices: $\mathbf{H} = (h_{k,n})$ and $\mathbf{\Phi} = (\varphi_{k,n})$ where $h_{k,n}^2$ and $\varphi_{k,n}$ denote respectively the propagation gain and the phase shift of the channel for signals transmitted on the n -th Tx RF chain and received on the k -th Rx RF chain. To refer to this channel, we adopt hereafter the notation $n \rightarrow k$. In addition, it is of interest to express the propagation gain for channel $n \rightarrow k$ as $h_{k,n}^2 = g_{k,n}^{\text{Tx}} \cdot l_{k,n} \cdot g_{k,n}^{\text{Rx}}$ to highlight the influence of the antennas directivity gains $g_{k,n}^{\text{Tx}}$, $g_{k,n}^{\text{Rx}}$ and the path loss $l_{k,n}$. Column vectors $\mathbf{s} = [s_1 \dots s_{N_t}]^T$ and $\mathbf{r} = [r_1 \dots r_{N_r}]^T$ denote the sent and received symbols. Envelope modulation is used at the transmitter and ED at the receiver.

A. Transmitter RF chain

The transmitter implements envelope modulation and the architecture of one of its RF chains is depicted in Fig. 2. Envelope modulation allows a simple implementation and an efficient use of power amplifiers. In this case, OOK appears to be a simple and efficient modulation scheme considering a non-coherent demodulation – see flash-signaling in [18]. On the n -th RF chain, we write $s_n(t) \in \mathbb{R}_{\geq 0}$ the modulating signal resulting from a rectangular pulse-shaping Π :

$$s_n(t) = \sum_{\tau \in \mathbb{Z}} s_n[\tau] \cdot \frac{\Pi\left(\frac{t}{T} - \tau - \frac{1}{2}\right)}{\sqrt{T}}, \quad t \in \mathbb{R}, \quad (1)$$

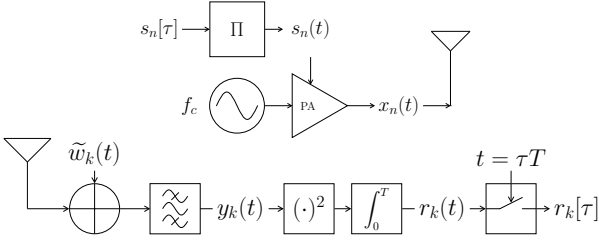


Fig. 2: Block diagram of one Tx-Rx chain

where $s_n[\tau]$ is the τ -th modulated symbol from constellation $\mathcal{C} = \{0, \sqrt{2}\}$ and T is the symbol duration. We have $\int_{\tau T}^{\tau T+T} |s_n(t)|^2 dt = s_n[\tau]^2$. The transmitted signal $x_n(t)$ at carrier frequency f_c is given by

$$x_n(t) = s_n(t) \cdot \sqrt{2} \cos(2\pi f_c t + \phi(t)), \quad (2)$$

where $\phi(t)$ is a stochastic process modeling a strong oscillator phase noise. The transmitter uses a single oscillator reference, common to all RF chains, hence the phase noise process $\phi(t)$ the same.

B. Propagation channel

Recent measurement campaigns have shown that sub-THz propagation channels are largely dominated by a single path, often the LoS direct path, which provides most of the energy contribution [19] [20]. This is due to the stronger channel sparsity at those frequencies, in particular in open or urban environment, and to the usage of highly directive antennas, sometimes at both transceiver sides. The indoor radio propagation channel between 126 GHz and 156 GHz has been characterized in [19] and in [20] through measurements. Otherwise, the modeling of sub-THz channels is addressed in [21] using deterministic ray-tracing. Accordingly, we assume in this paper a static LoS channel model.

C. Receiver RF chain

We detail here the receiver RF chains whose architecture is depicted in Fig. 2 for the k -th RF chain. The input of the k -th chain is the band-limited¹ signal $y_k(t)$, with bandwidth $B \geq 2/T$ centered around carrier frequency f_c . This signal is given by

$$y_k(t) = \sum_{n=1}^{N_t} h_{k,n} s_n(t) \sqrt{2} \cos(2\pi f_c t + \varphi_{k,n} + \phi(t)) + w_k(t), \quad (3)$$

where $w_k(t)$ is a band-limited continuous Gaussian process with spectral density N_0 , modeling the thermal noise. To simplify further derivations, it is convenient to use the quadrature representation of the real thermal noise $w_k(t)$. Accordingly, $w_k(t)$ is represented using the real baseband signals $w_{k,c}(t)$ and $w_{k,s}(t)$ in quadrature and modulated at f_c . It follows that

$$w_k(t) = w_{k,c}(t) \sqrt{2} \cos(2\pi f_c t) - w_{k,s}(t) \sqrt{2} \sin(2\pi f_c t), \quad (4)$$

¹The band-pass filter prevents the spectral folding of noise.

where $w_{k,c}(t)$ and $w_{k,s}(t)$ are band-limited continuous Gaussian process with spectral density $N_0/2$, $\varphi_{k,n} = d_{k,n} \cdot 2\pi f_c / c$ is the channel phase shift, with $d_{k,n}$ the propagation distance and c the light speed. The frequency down-conversion of the signal to baseband is achieved without impact of phase noise by squaring signal $y_k(t)$ and low-pass filtering it. That is

$$r_k(t) = \int_t^{t+T} y_k(u)^2 du, \quad (5)$$

with $f_c \gg 1/T$. For the k -th Rx RF chain, the τ -th received symbol $r_k[\tau] = r_k(\tau T)$ is obtained after integration and sampling, and can be expressed as

$$\begin{aligned} r_k[\tau] = & \sum_{n=1}^{N_t} \sum_{m=1}^{N_t} h_{k,n} s_n[\tau] \cdot h_{k,m} s_m[\tau] \cos(\varphi_{k,n} - \varphi_{k,m}) \\ & + 2 \int_{\tau T}^{\tau T+T} w_{k,c}(t) \cdot \sum_{n=1}^{N_t} h_{k,n} s_n(t) \cos(\varphi_{k,n} + \phi(t)) dt \\ & + 2 \int_{\tau T}^{\tau T+T} w_{k,s}(t) \cdot \sum_{n=1}^{N_t} h_{k,n} s_n(t) \sin(\varphi_{k,n} + \phi(t)) dt \\ & + \int_{\tau T}^{\tau T+T} w_{k,c}(t)^2 + w_{k,s}(t)^2 dt. \end{aligned} \quad (6)$$

Regarding this equation, the first line represents the energy of the received signals of the different Tx chains whereas the following lines express the contribution of thermal noise in the received symbols. Accordingly, we denote the energy of the received signals

$$E_k[\tau] = \sum_{n=1}^{N_t} \sum_{m=1}^{N_t} h_{k,n} s_n[\tau] h_{k,m} s_m[\tau] \cos(\varphi_{k,n} - \varphi_{k,m}). \quad (7)$$

It should be noted that E_k expresses the nonlinear interference between the different channels of the system. Finally, we denote by $2M = \lfloor 2BT \rfloor + 1$ the time-bandwidth concentration of received signals [22] [23]. The time-bandwidth concentration is used in this paper to derive the probability distributions of received symbols. These derivations are detailed in the Appendix. It can be shown that the received symbols are given by

$$r_k[\tau] = E_k[\tau] + \sqrt{2E_k[\tau]} \cdot w_k[\tau] + z_k[\tau], \quad (8)$$

where $w_k[\tau] \sim \mathcal{N}(0, \sigma_w^2)$ is a zero-mean Gaussian variable with variance $\sigma_w^2 = N_0 B / 2M$ and $z_k[\tau] \sim \sigma_w^2 / 2 \cdot \chi_{4M}^2$ is a chi-square distributed variable with $4M$ degrees of freedom. Furthermore, Eq. (8) defines the nonlinear MIMO channel of the considered transceiver. In the following section, we investigate the design of the detection algorithm related to this channel. From now on, the time index τ is disregarded for brevity.

III. DETECTION ALGORITHM DESIGN

A. Threshold detector (TD)

We present in this paragraph the commonly used *threshold detector* abbreviated by *TD*. When using a single-antenna transceiver, this criterion has been shown to be optimal for the detection of an OOK with an ED receiver, *i.e.* minimizing

the error probability [24]. This decision rule, defined upon a threshold comparison, can be expressed as

$$\hat{s}_k = \begin{cases} 0, & \text{if } r_k < \lambda_{\text{opt}}, \\ \sqrt{2}, & \text{otherwise.} \end{cases} \quad (9)$$

where λ_{opt} is the optimal threshold, and depends on $h_{k,k}^2$. This detector only requires the estimation of propagation gains $h_{k,k}^2$. The expression of λ_{opt} and its evaluation in practical systems are presented in [24]. Estimating symbol \hat{s}_k on the k -th RF receiver chain, this detection criterion demodulates symbols from the different antennas independently. Using the TD in multiple-antennas systems implies the modeling of the spatial interference between channels as noise. On the one hand, if the spatial interference is negligible, this decision rule provides an efficient and low-complexity demodulation scheme. On the other hand, for strongly correlated channels, this detector might not be able to estimate sent symbols. In this case, a joint demodulation of the received symbols over all antennas should be used. We propose in the following paragraph a joint demodulation algorithm which exploits spatial interference between channels as information to demodulate symbols on strongly correlated channels.

B. Maximum likelihood detector with Gaussian approximation (MLD-GA)

1) *Position to state-of-the art techniques:* Well-known for linear MIMO channels, the ML detector jointly demodulates the received symbols of the different RF chains. This detector uses spatial interference as information to demodulate symbols. It is important to point out unlike linear MIMO channels, the spatial interference between channels expressed in Eq. (7) is nonlinear due to the squaring of received signals such that traditional detectors cannot be used. Therefore, we derive here a sub-optimal detector corresponding to the nonlinear MIMO channel described in Eq. (8), using a Gaussian approximation approach. The proposed detector is further abbreviated by MLD-GA (Maximum Likelihood Detector with Gaussian Approximation). It should be mentioned that [10] also addresses the demodulation for a nonlinear MIMO system. In [10], the demodulation is based on amplitude detection of complex symbols resulting from a coherent receiver with a local oscillator. Consequently the demodulation proposed in [10] may be sensitive to phase impairments since the output of the matched-filter is subject to a penalty in signal-to-noise ratio (SNR) penalty, resulting from phase noise [25]. Conversely, we consider envelop extraction using energy detectors robust to phase noise. For this reason, the detector described in [10] cannot be used for the studied transceiver and differs from the proposed MLD-GA.

2) *Derivation of the MLD-GA decision rule:* For independent and equiprobable symbols, the ML decision rule is optimum, *i.e.* minimizes the error probability. It is defined upon the channel likelihood by

$$\hat{s} = \arg \max_{\mathbf{s} \in \mathcal{C}^{N_t}} p(\mathbf{r}|\mathbf{s}, \mathbf{H}, \mathbf{\Phi}). \quad (10)$$

With regard to the superposition of χ^2 and Gaussian distributions in Eq. (8), the detection criterion resulting from the

channel likelihood would be too complex to be evaluated in practical systems. To derive a decision rule with a simple implementation, we approximate the contribution of χ^2 -distributed noise, precisely z_k in Eq. (8), by its expected value. The variance of z_k being $2M \cdot \sigma_w^4$, this approximation is tight at high SNR. Received symbol r_k then follows a Gaussian distribution

$$r_k \sim \mathcal{N}(\mu_k, \sigma_k^2), \quad (11)$$

where the mean and variance are given by

$$\mu_k = E_k + 2M \cdot \sigma_w^2, \quad (12)$$

$$\sigma_k^2 = 2E_k \cdot \sigma_w^2 + \varepsilon. \quad (13)$$

Though the noise contribution is approximated as Gaussian, the interference between channels remains nonlinear as expressed by E_k in Eq. (7). The term ε is introduced to prevent discontinuity² in the detection criterion. Then, the joint probability density function is given by

$$p(\mathbf{r}|\mathbf{s}, \mathbf{H}, \mathbf{\Phi}) = \prod_{k=1}^{N_r} \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp\left(-\frac{|r_k - \mu_k|^2}{2\sigma_k^2}\right). \quad (14)$$

Finally, the detection decision rule can be written as

$$\hat{\mathbf{s}} = \arg \min_{\mathbf{s} \in \mathcal{C}^{N_t}} \sum_{k=1}^{N_r} \frac{|r_k - \mu_k(\mathbf{s})|^2}{\sigma_k(\mathbf{s})^2} + \ln(\sigma_k(\mathbf{s})^2). \quad (15)$$

This expression fully describes the proposed MLD-GA. The minimization space \mathcal{C}^{N_t} increases exponentially with the number of antenna N_t . Since OOK modulation is used, the modulation order remains small, $|\mathcal{C}| = 2$, and the complexity of the MLD-GA algorithm is not an issue. It must be emphasized that the MLD-GA requires the estimation of \mathbf{H} and $\mathbf{\Phi}$. The implementation of such estimation algorithm exceeds the scope of this paper and hence is not presented. In the following, we assume perfect knowledge of the channel at the receiver.

C. Neural Networks based Detector (NND)

We now propose a novel and original *neural networks based detector (NND)* to estimate the sent symbols. The motivation of using neural networks is twofold. First, as discussed in the introduction, the considered system communicates through a nonlinear MIMO, and neural networks are efficient to solve multi-variables non-linear problems. Second, in the case of the MLD-GA algorithm, the impact of noise on received symbols is assumed to be Gaussian, assumption we do not make using the NND. In addition, the NND does not explicitly need the propagation matrices \mathbf{H} and $\mathbf{\Phi}$ to estimate symbols, as it learns the channel features during the training phase.

1) *Architecture of the NND:* The NND is composed of multiple neural networks estimating the transmitted symbols. It is worth noticing that the NND uses one neural network per transmit antenna. Each of the N_t neural networks estimates a single transmitted symbol. The proposed architecture of the k -th neural network is depicted in Fig. 3 for the detection of

²Empirical results have shown that setting $\varepsilon = \sigma_w^2$ is an efficient choice to minimize the bit-error-rate.

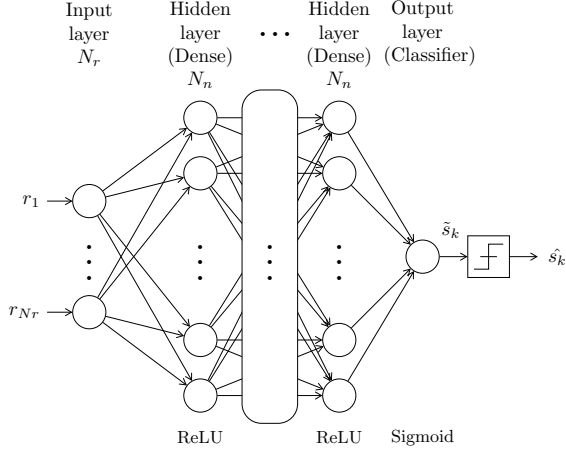


Fig. 3: Architecture of the k -th neural network of the NND

symbol s_k . This neural network uses fully connected hidden layers N_{hl} , each one composed of N_n neurons using rectified linear unit (ReLU) as activation function. The number of hidden layers is function³ of the number of transmit antennas. The input layer has N_r units, each one representing a received symbol r_k . Finally, a prediction \tilde{s}_k – homologous to the probability $Pr(s_k = 0|\mathbf{r})$ – is produced at the output layer with a sigmoid unit. Thus, we build a multi layer perceptron classifier in order to estimate the transmitted symbols s_k . Besides, it is worth mentioning that optimizing jointly the N_t neural networks is more complex, and we have not observed any performance improvement with respect to the parallel optimization. Therefore, we consider training each neural network independently in this paper. Eventually, the use of the NND is relevant for high-rate applications requiring parallel processing since sent symbols are estimated independently. Indeed, each neural network of the NND can be trained and provide inferences independently.

2) *Training of the NND*: The training of the NND is realized by transmitting some reference symbols known by the receiver. The set of weights is optimized during the training phase to maximize the detection performance of the NND. Each neural network, one per transmit antenna, is trained independently using an *Adam* (Adaptative Momentum estimation) optimizer [26]. The NND presents the advantage of having one loss function J_k to optimize per neural network. Since OOK modulation is exploited at the transmitter, it is pertinent to use a binary cross-entropy loss function described by

$$J_k = -\frac{1}{B_s} \sum_{\tau=1}^{B_s} \left[\frac{s_k[\tau]}{\sqrt{2}} \ln(\tilde{s}_k[\tau]) + \left(1 - \frac{s_k[\tau]}{\sqrt{2}}\right) \ln(1 - \tilde{s}_k[\tau]) \right] \quad (16)$$

where B_s stands for the batch size. In addition, it may be noted that the fixed property of the wireless link in the targeted application is particularly suited to machine learning. Indeed, a calibration phase must be realized once the system is deployed because the received symbols depends on \mathbf{H} and

³We note that the choice of these parameters is empirical and we have observed for $N_t \leq 8$ some relations between the NND parameters and the system parameters such as $N_{hl} = N_t/2$ and $N_n = N_t^2$.

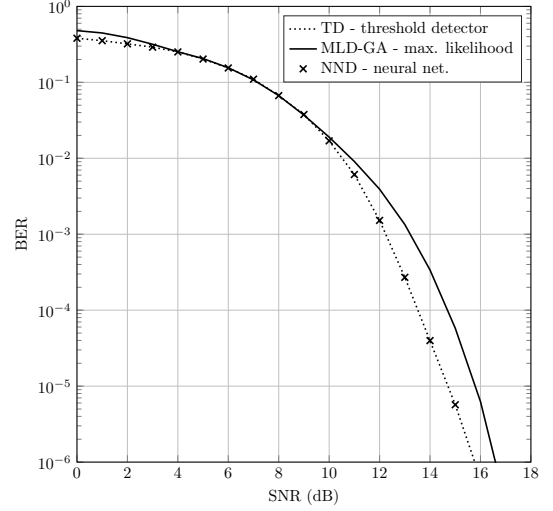


Fig. 4: Detection performance without spatial interference

Φ . Moreover, the channel is assumed to vary slowly or even be static. Therefore we only need to train the NND once on pilot symbols. The spectral efficiency loss due to pilot symbols becomes insignificant because of the large channel coherence time for the envisaged scenarios. A detailed description of the NND implementation parameters is outlined in the following section presenting the results of numerical simulations.

IV. PERFORMANCE ANALYSIS

A. Systems without spatial interference

To evaluate the performance of the detectors, we first investigate MIMO systems without spatial interference. The channels are perfectly spatially multiplexed, *i.e.* \mathbf{H} is diagonal. To implement the TD decision rule in practical systems, the threshold λ_{opt} has to be evaluated efficiently. Therefore, we use the expression of λ_{opt} proposed in [24] using a polynomial approximation.

Fig. 4 presents the results of numerical simulations for systems using OOK with $N_t = N_r$ and no spatial interference. The communication performance is expressed in terms of bit-error-rate (BER) as a function of the SNR defined by $h_{k,k}^2/\sigma_w^2$. For transceivers without spatial interference, it can be shown that the NND achieves the optimal detection performance given by the TD. We can remark that the TD and the NND present a performance gain in comparison to the MLD-GA decision rule. The performance loss of the MLD-GA results from the Gaussian approximation of the channel. Indeed, in the present case, considering a system without spatial interference equivalent to a single transmit antenna, the condition for the Gaussian approximation to be accurate, $\sigma_w^2 \ll E_k$, is not satisfied when transmitted symbol $s_k = 0$, since $E_k = 0$. Consequently, the MLD-GA is in this case sub-optimal. It is worth mentioning here that the MLD-GA is specifically designed for multi-antenna systems. The performance loss does not question the value of this detector, which is to be highlighted in the next paragraphs.

TABLE I: Simulation parameters

Parameters	Notation	Values
Carrier frequency	f_c	145 GHz
Symbol rate	$1/T$	1 GHz
Bandwidth	$B = 2/T$	2 GHz
Thermal noise	N_0	-174 dBm/Hz
Noise figure	N_f	10 dB
Antenna gain	g_0	32 dBi
Beam width	θ	3 °
Side lobe level	ϵ	-20 dB
Distance Tx - Rx	d_0	10 m

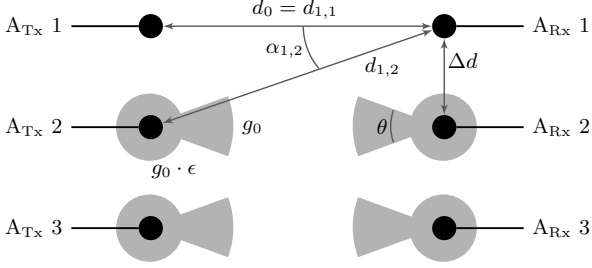


Fig. 5: Disposition of the antennas in the scenario

B. Scenario description: fixed indoor link in the D-band

We introduce here a realistic sub-THz scenario. The targeted application is a fixed indoor wireless link in the D-band. Table I outlines the main simulation parameters for this scenario. The considered system uses a uniform linear array (ULA) of antennas with $N_t = N_r = N$. The disposition of the antennas is depicted in Fig. 5. The specification of the antenna is extracted from [27] which describes the design of a high-gain antenna for the D-band based on transmit-arrays. We evaluate the propagation gain $h_{k,n}^2/\sigma_w^2$ for channel $n \rightarrow k$ using the link budget given by the Friis' transmission equation. With $P_{A_{Tx}}$ the transmit power per antenna, $G_{k,n}$ the antennas gain, and $d_{k,n}$ the propagation distance, the link budget is given by

$$\frac{h_{k,n}^2}{\sigma_w^2} = P_{A_{Tx}} G_{k,n} \left(\frac{c}{4\pi f_c d_{k,n}} \right)^2 \left(\frac{N_0 B}{2M} N_f \right)^{-1}. \quad (17)$$

We have $G_{k,n} = g_{k,n}^{Tx} \cdot g_{k,n}^{Rx}$ the product of the Tx and Rx antennas directivity gains for channel $n \rightarrow k$. We assume the commonly used sectorized antenna model illustrated in Fig. 5. The antenna directivity gain is then defined by

$$g(\alpha) = \begin{cases} g_0, & \text{if } |\alpha| < \frac{\theta}{2}, \\ g_0 \cdot \epsilon, & \text{otherwise,} \end{cases} \quad (18)$$

depending on the beam width θ , the beam offset angle to the main lobe α , the side lobe level ϵ ($0 < \epsilon \ll 1$) and the antenna gain g_0 . The considered scenario is symmetric and the beam of the k -th transmit antenna is aligned with the k -th receive one. This leads to $g_{k,n}^{Tx} = g_{k,n}^{Rx} = g(\alpha_{k,n})$ with $\alpha_{k,n}$ the beam offset angle for channel $n \rightarrow k$. Eventually, the channel matrix \mathbf{H} may be evaluated using Eq. (17) and Table I. The maximum propagation gain $h_{k,k}^2$ is achieved for any channel $k \rightarrow k$. For a transmit power per antenna $P_{A_{Tx}} = -30$ dBm, we have

$$\text{SNR} = \frac{h_{k,k}^2}{\sigma_w^2} \simeq 16.23 \text{ dB}. \quad (19)$$

For $k' \neq k$, the interference terms (off-diagonal elements) in matrix \mathbf{H} are approximately

$$\frac{h_{k,k'}^2}{h_{k,k}^2} \simeq \begin{cases} -0.01 \text{ dB}, & \text{if } |\alpha_{k,k'}| < \frac{\theta}{2}, \\ -40.02 \text{ dB}, & \text{otherwise.} \end{cases} \quad (20)$$

When $\Delta d \ll d_0$ the differences in path loss between channels are close to zero. Subsequently, the level of interference between two channels, either strong 0 dB or very low -40 dB, only results from the angle $\alpha_{k,k'}$. If the inter-antenna distance Δd is sufficiently large,

$$\Delta d \geq d^* = d_0 \tan\left(\frac{\theta}{2}\right), \quad (21)$$

the channels are almost perfectly spatially multiplexed. In this case, the interference results from side lobes of the antennas and is very low (< -40 dB). This corresponds closely to an ideal case and the system performance for any N equals the one described in Fig. 4 without interference. Since $d^* \simeq 26$ cm, the latter condition implies that the width ℓ of the array of antennas may be significantly large for practical implementation. The width of the ULA $\ell = d_A + (N - 1)\Delta d$, where $d_A = 5$ cm is the width of the antenna itself [27]. By way of illustration, for $N = 8$ we have $\ell > 1.8$ m. To reduce the width of the transceiver, we now consider a smaller inter-antenna distance satisfying

$$d^* \frac{2}{\kappa - 1} > \Delta d \geq d^* \frac{2}{\kappa + 1}, \quad (22)$$

where $\kappa \in \{3, 5, \dots, 2N - 1\}$. In this case, the main lobe of one transmit antenna beam enlightens multiple receive antennas, namely up to κ . Put differently, κ denotes the maximum number of transmitted symbols strongly interfering on a receive antenna. Hence, the larger the value of κ , the stronger the spatial interference between channels. By means of illustration, the channel gain matrix \mathbf{H} for $N = 4$ and $\kappa = 3$ may be accurately approximated⁴ as follows

$$\mathbf{H} \simeq h_{1,1} \cdot \begin{bmatrix} 1 & 1 & \rho & \rho \\ 1 & 1 & 1 & \rho \\ \rho & 1 & 1 & 1 \\ \rho & \rho & 1 & 1 \end{bmatrix}, \quad (23)$$

where ρ is the residual interference due to side lobes of the antennas with $\rho^2 = -40$ dB. There are κ diagonals whose elements are 1 which equals the maximum number of interfering symbols on a receive antenna. We use hereafter $\kappa = 1$ to denote the multiplexed case corresponding to Eq. (21). Eventually, parameter κ enables us to quantify the level of spatial interference.

C. Systems with strong spatial interference

We now evaluate the influence of the spatial interference on the system performance, using parameter κ . In these simulations, the propagation gain and the phase shift matrices are not approximated, but are exactly computed from the scenario description and simulation parameters.

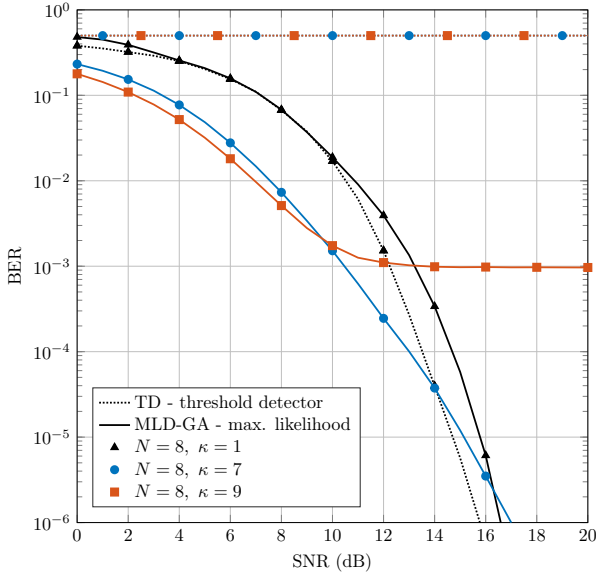


Fig. 6: Influence of the level of spatial interference (parameter κ) on the system performance. Solid lines are for the MLD-GA while dashed line for the threshold detector.

1) *MLD-GA vs. TD*: The results of numerical simulations comparing the detection performance of the MLD-GA and the TD are depicted in Fig. 6. The BER performance is assessed for $N = 8$ and different values of κ . First, it must be emphasized that the MLD-GA is essential to communicate on strongly correlated channels. As expected for $\kappa > 1$ the TD cannot demodulate sent symbols and presents a BER of $1/2$. Nevertheless, we can remark that if the spatial interference is too strong, *e.g.* $\kappa = 9$ in Fig. 6, the BER reaches an error floor. Simulation results also show that in the moderate SNR regime, transceivers with large values of κ , *i.e.* strong spatial interference, may demonstrate lower BER than the multiplexed case $\kappa = 1$. With channel coding, such property may be beneficial for configurations with κ large. If the BER is low enough, the waterfall feature of the decoding algorithm may be exhibited at a lower SNR. Subsequently, setting κ large is interesting to achieve low error rate communications with the combination of a channel coding while reducing the width of the transceiver. The properties exhibited for $N = 8$ in Fig. 6 also hold for different values of N .

2) *MLD-GA vs. NND*: This paragraph presents the communication performance of the NND and the MLD-GA algorithms on strongly correlated LoS channels. Table II presents the parameters of the NND architecture for the different system configurations. The training dataset contains about 10^6 symbols when using a 8×8 MIMO system. These symbols are used to train the 8 neural networks estimating each one a transmitted symbol. The neural networks are trained independently using 50 to 150 epochs. For the simulations presented

in this paragraph, NND is trained on Graphics Processing Unit (GPU) processor and inferred on Central Processing Unit (CPU) processor. However, in a practical system, for fixed indoor links, a calibration phase is needed using pilot symbols. These pilot symbols could be sent to a cloud to perform the training phase for example. Then, the NND would be inferred in an embedded system using an ASIC (Application-Specific Integrated Circuit) or a FPGA (Field-Programmable Gate Array) circuit. Here the tuning of the NND parameters is empirical, yet we can admit that increasing the number of antennas induces an increase of the number of hidden layers and neurons. The results of numerical simulations are depicted in Fig. 7 and Fig. 8. The BER performance is assessed for $N = 4$ and $N = 8$ and different values of κ , *i.e.* different levels of spatial interference. First, we can notice that, for $\kappa = 1$ with any number of antennas N , the performance equals the one without interference described in Sec. IV-A. It can be noticed in Table II that there is no hidden layer for the case without interference. Indeed, only one neuron with sigmoid output is sufficient. Second, for $N = 4$ with spatial interference, it can be observed that the MLD-GA and the NND present similar detection performance. Though, the NND demonstrates slight performance gains. Nonetheless, it should be noted that if the spatial interference is too strong, *e.g.* $\kappa = 5$ for $N = 4$ or $\kappa = 9$ for $N = 8$, the BER reaches an error floor. Third, for $N = 8$, we notice that the system performance of the NND is close to the one of the MLD-GA if the spatial interference is not too strong. With strong interference, $\kappa = 9$ and $N = 8$, using the NND leads to a significant performance loss. Moreover, at low SNR, the MLD-GA uses a Gaussian approximation whereas the NND does not. Therefore, it explains why the NND is generally better than the MLD-GA. However, this property is not verified for $\kappa = 9$ due to high level of spatial interference. The current NND architecture has difficulties to cope with high interference level which explains the lack of performance for the case $\kappa = 9$. Further investigations regarding the architecture and the learning are expected to reduce the performance loss. It should also be mentioned that increasing the number of neurons in that case does not induce performance gain, meaning that with this architecture we may not expect better performance for $\kappa = 9$ and $N = 8$.

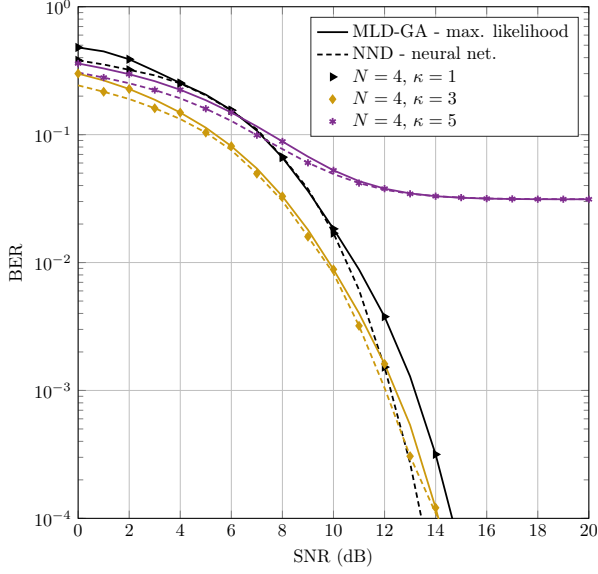
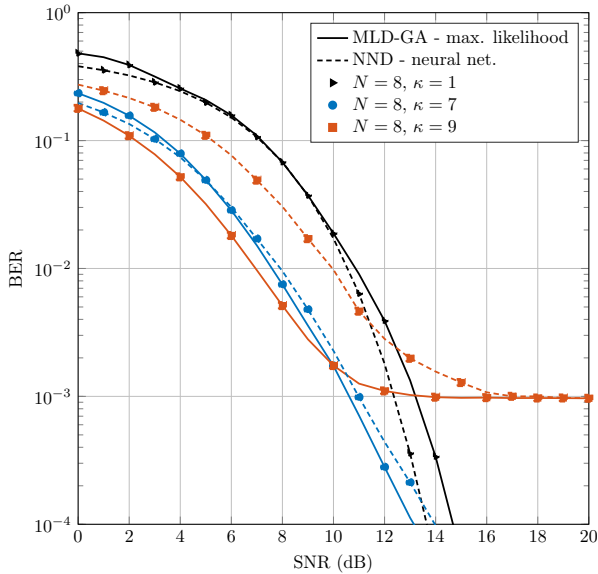
D. Discussions

We conclude this section on the analysis of the system performance analysis by discussing some properties of the transceiver. First, and similarly to linear MIMO systems, the considered system benefits from the spatial diversity. Sent symbols of a transmit-antenna may be received on several receive-antennas. This property thus improves the robustness to thermal noise. It explains the performance gains achieved by systems with strong interference in the moderate SNR regime in comparison to the multiplexed case, for instance as shown in Fig. 6. In addition to diversity, the studied system is also subject to ambiguity. Since the communication channel is a nonlinear MIMO channel, different transmitted MIMO symbols s may lead to similar received observations r . For this

⁴Since the differences in path loss is less than 0.02 dB, the adjacent channels interference and the residual side lobes interference can be respectively approximated to 1 and a constant ρ .

TABLE II: NND parameters

Parameters	MIMO no interference	4×4 MIMO	8×8 MIMO
Hidden layers (HL)	0	2	4
Neurons/HL	—	16	64
Batch size	32	32	256
Dataset size	10^5 symbols	10^6 symbols	
Epochs	50	150	150
Optimizer	Adam		
Learning rate	0.001		
Loss function	Binary cross-entropy Eq. (16)		
Activation functions	ReLU and Sigmoid		

Fig. 7: Performance of the MLD-GA and the NND for $N = 4$. Solid lines are for the MLD-GA while dashed lines for the NNDFig. 8: Performance of the MLD-GA and the NND for $N = 8$. Solid lines are for the MLD-GA while dashed lines for the NND

reason, the BER of system configurations with strong spatial interference reaches an error floor, *e.g.* $\kappa = 5$ for $N = 4$ in Fig. 7 or $\kappa = 9$ for $N = 8$ in Fig. 8. Subsequently, we claim that the level of spatial interference, and hence the inter-antenna spacing Δd , is directly related to a trade-off between diversity and ambiguity. Second, it is also worth mentioning that significant differences in channel qualities exist. The different spatial streams – corresponding to a pair of aligned transmit and receive antennas – are not subject to the same interference. As illustrated by Eq. (23), the numbers of symbols strongly interfering is larger for the receive antennas in the middle of the ULA than for the antennas on the extremities. Consequently, the interference is stronger for the antennas in the middle of the ULA than the ones on the extremities. We will see in the next section that this property can be exploited by the channel coding scheme to enhance the system performance by adaptive coding rate selection. In all our numerical simulations, we assume that the antennas are perfectly aligned. In case of misalignment, the entire system model and algorithms remain valid. The misalignment effect will be captured by the channel estimation scheme for the MLD-GA and by the learning mechanism for the NND.

Finally, our results demonstrate that spatial multiplexing with non-coherent sub-THz transceivers can be realized on strongly correlated LoS channels. MIMO systems with ED receivers hence offer a valuable solution to achieve high rate communications in sub-THz frequencies. Since the analysis is carried out with some approximations, in particular on the antenna model, the exactitude of the conclusions could be questioned. Nevertheless, the next section proposes to improve the performance analysis by considering and discussing some practical implementation issues.

V. IMPLEMENTATION CONSIDERATIONS

We have previously shown that low-complexity low-power transceivers using MIMO systems with energy detection receivers can be implemented within sub-THz frequencies. In this section, we first evaluate the system performance with the real antenna radiation pattern from [27]. Second, we consider the integration of a forward error correction (FEC) scheme as channel coding. Third, the practical implementation of the proposed detection schemes is discussed.

A. Performance with the real antenna gain

Previously, we have based the performance analysis upon the commonly used sectorized antenna model. The sectorized model is relevant for its simple analytic expression. However,

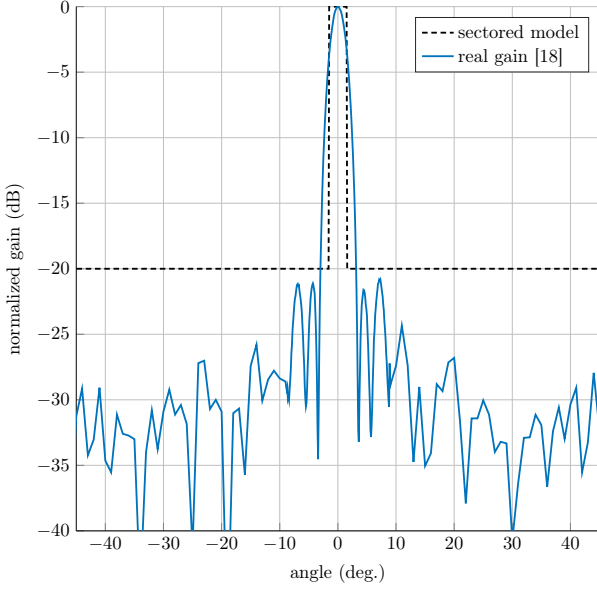


Fig. 9: Antenna gain measured in [27] (normalized to 32 dBi)

the accuracy of the analysis can be improved by considering the real antenna radiation pattern measured and published in [27]. Fig. 9 depicts the real antenna radiation pattern and the sectored model. The assessment of communication performance with the real antenna gain, in comparison to the sectored model, is presented in Fig. 10 for $N = 8$ and using the MLD-GA and the NND. It can be observed that the BER performance of system configurations with strong interference is deteriorated when the real antenna gain is considered. Specifically, the performance are worse at low SNR for the real antenna radiation pattern. The performance loss can be explained by a loss of diversity due to lower side lobes. However, it should be emphasized that the error floor is removed. With the real radiation pattern, sent symbols are received on multiple antennas but with lower energy than with the sectored pattern. Nevertheless, and conversely to the sectored model, it can be noted that ambiguity is removed and thus no error floor is observed with the real antenna radiation pattern, see the configuration $N = 8$, $\kappa = 9$. In addition, it should be emphasized that the MLD-GA and the NND demonstrate similar demodulation performance in the case of the real antenna gain. We conclude from these results that the sectored antenna model is an efficient but mostly optimistic model to describe practical systems. In the following, any further performance analysis is based on the real radiation pattern of the antenna directivity gain.

B. Performance with channel coding

We have previously demonstrated the influence of the level of spatial interference (parameter κ) on the system performance. It can be observed that the waterfall BER performance is improving with increasing κ . However, when κ increases too much, the BER may reach an error floor (e.g., $\kappa = 9$, the BER reaches an error floor at 10^{-3}). Yet, for a coded system such an error floor may be lowered or even removed by the FEC scheme. Therefore, it is interesting to consider

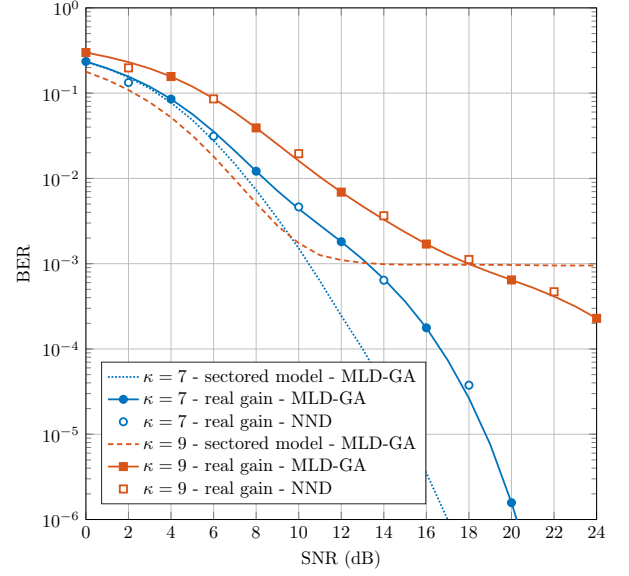


Fig. 10: Performance for $N = 8$ with the real antenna gain

the integration of a FEC scheme to achieve channel coding gain and low error rates. However, implementing the FEC, and in particular its decoder, may entail a significant complexity and power consumption. To achieve a low-complexity low-power transceiver, we propose here to use a BCH code. The considered FEC scheme is a BCH code with a packet size of 63 bits and a coding rate ranging from 0.4 to 0.9. We consider a hard-input, syndrome-based, half the minimum distance bounded decoding algorithm. It should be mentioned that the key features of this code are a low-complexity implementation and a low-power consumption [28]. In addition, with regard to the short packet size, this code has a low-latency decoder. These features appear to be highly relevant for the scenario investigated in this paper. The considered transceiver architecture with the integration of a channel coding is presented in Fig. 11. Multiple FEC schemes are used and the coding rate can be adapted to the channel quality of the receive antenna. Since the channel is assumed to be static, channel coding with adaptive rate is implemented only to adapt the coding rate to the receive antenna. Indeed, the channel presents significant differences in terms of quality depending on the receive antenna. Receive antennas in the middle of the ULA are subject to stronger interference than the ones on the extremities. For this reason, adapting the coding rate to the receive antenna enables us to capitalize on the latter property to further enhance demodulation performance. The system architecture in Fig. 11 is particularly interesting as it also maintains a high degree of parallelism.

Fig. 12 presents the achievable rates as function of E_b/N_0B for systems with a BCH code such that the BER is below 10^{-6} . The BCH code is implemented with a coding rate ranging from 0.4 to 1 and a channel decoder based on the hard decisions produced by the MLD-GA. Numerical results have been obtained through Monte-Carlo simulations with the real antenna radiation pattern. First, it should be remarked that integrating a FEC scheme enables to achieve a significant

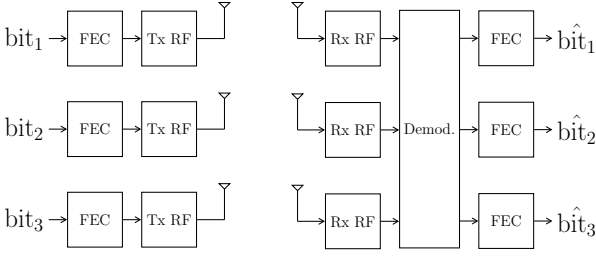


Fig. 11: System architecture integrating a FEC scheme

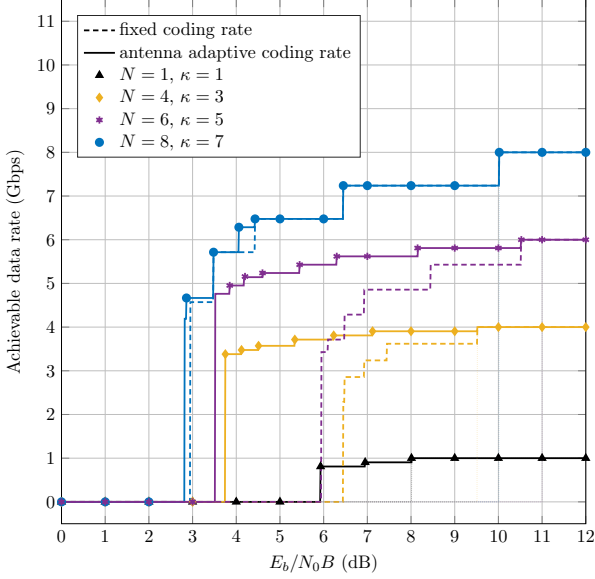


Fig. 12: Achievable data rate with a BCH code and the real antenna gain. The dashed lines are for a scheme with fixed coding rate and the solid lines for the achievable data rate with an adaptive coding rate strategy

channel coding gain. Second, it can be noticed that the adaption of the coding rate to the receive antenna leads to performance gains in comparison to setting a fixed coding rate for all antennas. In particular, we can see that for $N = 4$ and $N = 6$ the performance gains are larger than 2 dB.

To present the results of Fig. 12 differently, we propose in Table III a synthesis of the system performance and parameters for different number of antennas. For all system configurations, the system bandwidth is $B = 2$ GHz, the distance Tx-Rx is $d_0 = 10$ meters, and the coding rate of the BCH is 0.9. Though the system performance is evaluated with the MLD-GA, similar results are expected using a demodulation based on the NND. Further, a channel bonding scheme, aggregating several sub-bands, could increase the throughput and allow to benefit from the large available free spectrum offered in sub-THz bands. It can be concluded that MIMO systems using ED receivers may achieve high rate communications in sub-THz bands with low-power and low-complexity RF architectures. Performance of coded systems could be further improved by considering longer packet length, soft-decision channel decoding, or capacity-achieving codes, *e.g.* a polar code, yet at the detriment of complexity.

C. Comparison of the proposed detection algorithms

This section provides an overview of practical aspects of MLD-GA and NND techniques, to understand their respective benefits and drawbacks.

1) *Channel estimation*: The first difference between the MLD-GA and NND is the following. To perform its decision rules, the MLD-GA algorithm requires an explicit knowledge of H , σ_w and Φ , and hence, also the design of a channel estimation algorithm. In contrast, the NND is able to learn the channel features, and implicitly the channel matrices, during the training phase in order to demodulate the received symbols. The transmission of reference symbols is required for both detection algorithms which results in a spectral efficiency loss.

2) *Algorithmic and implementation complexity*: The computational complexity of MLD-GA and NND algorithms can be estimated. Yet, from an implementation perspective, it is delicate to draw conclusions based on the computational complexity only, as these algorithms lie in different paradigms. The MLD-GA algorithm is a common detection method. Although the complexity of the MLD-GA is $\mathcal{O}(|\mathcal{C}|^N)$ and increases exponentially with the number of antennas, the decision rule only requires the evaluation of simple weighted Euclidean distances. Also, the complexity of the MLD-GA depends on the order of the modulation scheme. Using an OOK modulation with $|\mathcal{C}| = 2$, the resulting complexity is $\mathcal{O}(2^N)$ which is reasonable for practical implementation. The implementation of the MLD-GA in practical systems would likely use a common digital signal processor. In contrast, the computational complexity of the NND depends mainly on the size of the used neural networks. Neural networks present high computational cost because they perform matrix multiplication. Hence, for a fully connected layer (dense), the order of the complexity is $\mathcal{O}(N_{in} \times N_n \times N_{out})$. In our case, for a 8×8 MIMO system, it leads to $\mathcal{O}(N_t^4)$, thus resulting in a computational complexity that may significantly exceed the one of MLD-GA. Regarding strict algorithm complexity, NND is more complex than the MLD-GA w.r.t. the demapping task. However, NND allows to perform “inner” channel estimation, which is complex to perform in our system model, and would be able to cope with additional non linear effects such as power amplifier non linearities. The proposed NND allows to show that with simple data, we can design a solution able to perform CSI estimation, detection and additional corrections. Besides, implementing neural network computations relies on very optimized algorithms [29] benefiting from cache usage, parallelism and shared memories [30]. For instance, fully connected layers may benefit from data and model parallelism, and pipelining; while in each layer, operations can be parallelized on either GPU or CPU. Besides, increasing the number of layers would drastically increase the complexity. Thereby, in our case, increasing the number of antennas will also increase the complexity of the NND. The complexity of both training and inference stages should be differentiated. Indeed, the training stage involves many data and consuming resources whereas the inference stage could be quite simple. Regarding the inference stage, it could be envisaged to quantize the NND weights to reduce

TABLE III: Synthesis of the main system parameters and key performance indicators

Carrier frequency	f_c	145 GHz			
Bandwidth	B	2 GHz			
Propagation distance	d_0	10 m			
Antenna gain	g_0	32 dBi			
Number of antennas	N	1	4	6	8
Throughput	$N/T \cdot 0.9$	0.9 Gbps	3.6 Gbps	5.4 Gbps	7.2 Gbps
Power by antenna	$P_{A_{Tx}}$	-31.8 dBm	-31.2 dBm	-30.4 dBm	-32.3 dBm
Width of the ULA	ℓ	5 cm	44 cm	50 cm	55 cm
Inter-antenna distance	Δd	\emptyset	13 cm	9 cm	7 cm

resource usage and enhance energy efficiency. One may also consider pruning the neural networks to reduce the complexity, *i.e.* deactivate neurons with low valued weights and dead ones. Besides, the implementation of neural networks is currently a largely investigated research topic. Thus, implementing neural networks on dedicated hardware such as GPU, FPGA and ASIC can significantly decrease computation time regarding the chosen neural network architecture – readers may refer to [31] and [32]. It can be noted that in our system, GPU would be used for the calibration phase and a more energy efficient hardware for inference such as FPGA or ASIC. Moreover, an analog implementation of the decoder integrated to the analog front end, through spiking neural network [33] or the use of programmable discrete components [34] could help to reduce complexity and power consumption, especially to deal with very high data rate. However as we deal with digital transmission a one bit ADC is still necessary.

Last, it should be emphasized that in this work, for the contemplated applications such as device to device communication or indoor backhaul, it appears to be challenging to implement the proposed detectors in systems with $N \geq 16$ due to: i) the width of the antenna array, and ii) the complexity of the detector. However larger antenna system could be envisaged for other applications, *e.g.* outdoor backhaul link.

3) *Transceiver RF nonlinearities*: Eventually, it is interesting to mention that RF components of the transceiver might present additional nonlinearities, *e.g.* quantization, RF power amplifier. In particular, envelope detectors based on diodes may present non-ideal square law response such that $X_{out} \propto X_{in}^\alpha$ with $\alpha < 2$ [35]. It is expected that the NND might learn these channel nonlinearities and still demodulates symbol efficiently. The MLD-GA does not consider any other nonlinearities, such that it might be sensitive to these impairments and might require additional modeling. Nevertheless, it is of practical interest to characterize the detector robustness to imperfect RF components.

VI. CONCLUSION

We have investigated the design of MIMO systems with ED receivers for future applications in sub-THz bands. First, the system model has been described by characterizing the sub-THz channel and the RF architectures of the transmitter and receiver. Next, we have proposed two detection algorithms: i) the MLD-GA, derived from the ML decision rule for the studied nonlinear MIMO channel using a Gaussian approximation; ii) the NND, a detector based on the use of neural networks.

Subsequently, a realistic scenario modeling an indoor wireless link has been considered to assess the communications performance. Simulation results have proved that low error rate communications can be achieved on strongly correlated LoS channels using the proposed detection schemes. The two proposed detectors present similar demodulation performance, yet their implementations in practical systems are very different. Moreover, we have shown that integrating a low-complexity channel coding scheme leads to valuable performance gains in terms of achievable data rate. In conclusion, our results demonstrate that spatial multiplexing with non-coherent sub-THz transceivers can be realized on strongly correlated LoS channels. Ultimately, the spectral efficiency of non-coherent communication systems using sub-THz bands can be efficiently increased using MIMO systems and ED receivers with low-complexity and low-power RF architectures.

Besides, it is worth mentioning that the presented techniques and results are relevant to applications beyond sub-THz communications such as visible light communications or optical systems. Visible light communication systems commonly implement intensity modulation and detection. The nonlinear interference between channels in these systems is also a major challenge to realize spatial multiplexing. The proposed detectors could be easily adapted to visible light communication systems in order to achieve spatial multiplexing.

ACKNOWLEDGMENTS

This work has been supported by CPS4EU project, which has received funding from the ECSEL Joint Undertaking (JU) under grant agreement 826276.

APPENDIX

We derive here the probability distribution of received symbols expressed by Eq. (24). In detail, this paragraph intends to evaluate the distributions of the integrals in Eq. (24) expressed in the considered nonlinear MIMO channel of Eq. (7) by

$$\sqrt{2E_k[\tau]} \cdot w_k[\tau] = 2 \int_{\tau T}^{\tau T+T} w_{k,c}(t) S_{k,c}(t) + w_{k,s}(t) S_{k,s}(t) dt \quad (24)$$

$$z_k[\tau] = \int_{\tau T}^{\tau T+T} w_{k,c}(t)^2 + w_{k,s}(t)^2 dt. \quad (25)$$

The following notations are used

$$S_{k,c}(t) = \sum_{n=1}^{N_t} h_{k,n} s_n(t) \cos(\varphi_{k,n} + \phi(t)),$$

$$S_{k,s}(t) = \sum_{n=1}^{N_t} h_{k,n} s_n(t) \sin(\varphi_{k,n} + \phi(t)). \quad (26)$$

It should be mentioned that in Eq. (8) the terms $w_k[\tau]$ and $z_k[\tau]$ respectively denote the mixed signal-noise contribution and the squared noise contribution in the channel. Under a strong oscillator phase noise assumption, we obtain

$$\int_{\tau T}^{\tau T+T} S_{k,c}(t)^2 dt = \int_{\tau T}^{\tau T+T} S_{k,s}(t)^2 dt = E_k[\tau]/2. \quad (27)$$

The strong oscillator phase noise assumption corresponds to $\int_T \cos(\phi(t))^2 dt = \frac{T}{2} + o(1)$. This assumption is verified for a fast varying phase noise (strong phase impairments). Nevertheless, if not the case, nothing but the variance of the mixed signal-noise term $w_k[\tau]$ is different, which has no further impact on the design of the detection algorithm. For this reason, this assumption results in no loss of generality. Signals $S_{k,c}(t)$, $S_{k,s}(t)$, $w_{k,c}(t)$ and $w_{k,s}(t)$ are band-limited and finite energy signals. With an observation of duration T and a band B , these signals lie in a signal space of dimension $2M = \lfloor 2BT \rfloor + 1$, see [22]. Hence these signals can be decomposed onto an orthonormal basis $\psi = \{\psi_i\}_{1 \leq i \leq 2M}$ as follows

$$S_{k,c}(t) = \sum_{i=1}^{2M} S_{k,c}^i \cdot \psi_i(t), \quad S_{k,s}(t) = \sum_{i=1}^{2M} S_{k,s}^i \cdot \psi_i(t), \quad (28)$$

$$w_{k,c}(t) = \sum_{i=1}^{2M} w_{k,c}^i \cdot \psi_i(t), \quad w_{k,s}(t) = \sum_{i=1}^{2M} w_{k,s}^i \cdot \psi_i(t). \quad (29)$$

This decomposition enables us to derive the probability distribution of received symbols. It follows from Eq. (26) that the scalar coefficients of the decomposition verify

$$\sum_{i=1}^{2M} (S_{k,c}^i)^2 = \sum_{i=1}^{2M} (S_{k,s}^i)^2 = \frac{E_k[\tau]}{2}, \quad (30)$$

$$\mathbb{E} \left[\sum_{i=1}^{2M} |w_{k,c}^i|^2 \right] = \mathbb{E} \left[\sum_{i=1}^{2M} |w_{k,s}^i|^2 \right] = \frac{N_0 B}{2}, \quad (31)$$

where $\mathbb{E}[\cdot]$ is the expectation operator. Coefficients $w_{k,c}^i$ and $w_{k,s}^i$ are thus zero-mean Gaussian variables with variance $\sigma_w^2/2$. Let us recall that $\sigma_w^2 = N_0 B/2M$. By definition of basis ψ , $\int_T \psi_i(t) \psi_j(t) dt = \delta_{ij}$. We are now in a position to express the distributions of the integrals in Eq. (24). First, for the mixed signal-noise contribution, the decomposition of the signals on basis ψ leads us to

$$\int_{\tau T}^{\tau T+T} w_{k,c}(t) S_{k,c}(t) dt = \sum_{i=1}^{2M} \sum_{j=1}^{2M} w_{k,c}^i S_{k,c}^j \int_{\tau T}^{\tau T+T} \psi_i(t) \psi_j(t) dt. \quad (32)$$

$$\int_{\tau T}^{\tau T+T} w_{k,c}(t) S_{k,c}(t) dt = \sum_{i=1}^{2M} w_{k,c}^i S_{k,c}^i \sim \mathcal{N} \left(0, \frac{E_k[\tau]}{4} \sigma_w^2 \right). \quad (33)$$

Using similar derivations to evaluate $\int_T w_{k,s}(t) S_{k,s}(t) dt$, we obtain

$$\int_{\tau T}^{\tau T+T} w_{k,c}(t) S_{k,c}(t) + w_{k,s}(t) S_{k,s}(t) dt \sim \mathcal{N} \left(0, \frac{E_k[\tau]}{2} \sigma_w^2 \right). \quad (34)$$

Second, for the squared noise contribution, we can express the integral as follows

$$\int_{\tau T}^{\tau T+T} w_{k,c}(t)^2 = \sum_{i=1}^{2M} \sum_{j=1}^{2M} w_{k,c}^i w_{k,c}^j \int_{\tau T}^{\tau T+T} \psi_i(t) \psi_j(t) dt,$$

$$\int_{\tau T}^{\tau T+T} w_{k,c}(t)^2 = \sum_{i=1}^{2M} (w_{k,c}^i)^2 \sim \sigma_w^2 \cdot \chi_{2M}^2. \quad (35)$$

With an identical reasoning on $\int_T w_{k,s}(t)^2 dt$, it finally appears that

$$z_k[\tau] = \int_{\tau T}^{\tau T+T} w_{k,c}(t)^2 + w_{k,s}(t)^2 dt \sim \frac{\sigma_w^2}{2} \cdot \chi_{4M}^2 \quad (36)$$

In summary, the received symbols are given by

$$r_k[\tau] = E_k[\tau] + \sqrt{2E_k[\tau]} \cdot w_k[\tau] + z_k[\tau], \quad (37)$$

where $w_k[\tau] \sim \mathcal{N}(0, \sigma_w^2)$ and $z_k[\tau] \sim \sigma_w^2/2 \cdot \chi_{4M}^2$.

REFERENCES

- [1] S. Bicaïs, J.-B. Doré, and V. Savin, "Design of MIMO Systems using Energy Detectors for Sub-TeraHertz Applications," in *2020 IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Sept 2020, submitted.
- [2] T. S. Rappaport, Y. Xing, O. Kanhere, S. Ju, A. Madanayake, S. Mandal, A. Alkhatieb, and G. C. Trichopoulos, "Wireless Communications and Applications Above 100 GHz: Opportunities and Challenges for 6G and Beyond," *IEEE Access*, vol. 7, pp. 78 729–78 757, 2019.
- [3] J.-B. Doré, Y. Corre, S. Bicaïs, J. Palicot, E. Faussurier, D. Kténas, and F. Bader, "Above-90GHz Spectrum and Single-Carrier Waveform as Enablers for Efficient Tbit/s Wireless Communications," in *25th International Conference on Telecommunications (ICT'2018)*, Saint-Malo, France, Jun. 2018.
- [4] M. Voicu, D. Pepe, and D. Zito, "Performance and Trends in Millimetre-Wave CMOS Oscillators for Emerging Wireless Applications," *International Journal of Microwave Science and Technology*, vol. 2013, p. 6, 2013.
- [5] J.-L. Jimenez Gonzalez, C. Dehos, and N. Casisau, "Channel bonding transceivers for 6G future networks," in *2020 6G Summit*, 2020.
- [6] S. Bicaïs and J.-B. Doré, "Design of Digital Communications for Strong Phase Noise Channels," *IEEE Open Journal of Vehicular Technology*, vol. 1, pp. 227–243, 2020.
- [7] R. Combes and S. Yang, "An Approximate ML Detector for MIMO Channels Corrupted by Phase Noise," *IEEE Transactions on Communications*, vol. 66, no. 3, pp. 1176–1189, 2018.
- [8] J. Park, S. Kang, S. V. Thyagarajan, E. Alon, and A. M. Niknejad, "A 260 GHz fully integrated CMOS transceiver for wireless chip-to-chip communication," in *2012 Symposium on VLSI Circuits (VLSIC)*, June 2012, pp. 48–49.
- [9] L. Jing, E. De Carvalho, P. Popovski, and A. O. Martinez, "Design and Performance Analysis of Noncoherent Detection Systems With Massive Receiver Arrays," *IEEE Transactions on Signal Processing*, vol. 64, no. 19, pp. 5000–5010, Oct 2016.

- [10] G. K. Psaltopoulos and A. Wittneben, "Diversity and spatial multiplexing of MIMO amplitude detection receivers," in *2009 IEEE 20th International Symposium on Personal, Indoor and Mobile Radio Communications*, Sep. 2009, pp. 202–206.
- [11] T. O'Shea and J. Hoydis, "An Introduction to Deep Learning for the Physical Layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, 2017.
- [12] H. Huang, S. Guo, G. Gui, Z. Yang, J. Zhang, H. Sari, and F. Adachi, "Deep Learning for Physical-Layer 5G Wireless Techniques: Opportunities, Challenges and Solutions," *IEEE Wireless Communications*, vol. 27, no. 1, pp. 214–222, 2020.
- [13] N. Farsad and A. Goldsmith, "Neural network detection of data sequences in communication systems," *IEEE Transactions on Signal Processing*, vol. 66, no. 21, pp. 5663–5678, 2018.
- [14] N. Samuel, T. Diskin, and A. Wiesel, "Deep MIMO Detection," in *2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2017, pp. 1–5.
- [15] N. Samuel, T. Diskin, and A. Wiesel, "Learning to Detect," *IEEE Transactions on Signal Processing*, vol. 67, no. 10, pp. 2554–2564, 2019.
- [16] M. Khani, M. Alizadeh, J. Hoydis, and P. Fleming, "Adaptive Neural Signal Detection for Massive MIMO," *IEEE Transactions on Wireless Communications*, vol. 19, no. 8, pp. 5635–5648, 2020.
- [17] A. Elrharras, R. Saadane, M. Wahbi, and A. Hamdoun, "Hybrid architecture for spectrum sensing algorithm based on energy detection technique and artificial neural networks," in *2014 5th Workshop on Codes, Cryptography and Communication Systems (WCCCS)*, 2014, pp. 40–44.
- [18] S. Verdú, "Spectral efficiency in the wideband regime," *IEEE Trans. Information Theory*, vol. 48, pp. 1319–1343, 2002.
- [19] L. Pomietcu and R. D'Errico, "Characterization of Sub-THz and mmWave Propagation Channel for Indoor Scenarios," in *12th European Association on Antennas and Propagation (EurAAP 18)*, Apr 2018, pp. 1–4.
- [20] T. Xing and T. S. Rappaport, "Propagation Measurement System and Approach at 140 GHz—Moving to 6G and Above 100 GHz," in *2018 IEEE Global Communications Conference (GLOBECOM)*, Dec 2018.
- [21] G. Gougeon, Y. Corre, and M. Z. Aslam, "Ray-based Deterministic Channel Modelling for sub-THz Band," in *2019 IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Sep. 2019.
- [22] P. Dollard, "On the time-bandwidth concentration of signal functions forming given geometric vector configurations," *IEEE Transactions on Information Theory*, vol. 10, no. 4, pp. 328–338, October 1964.
- [23] J. Proakis, *Digital Communications 5th Edition*, ser. McGraw-Hill series in electrical and computer engineering : communications and signal processing. McGraw-Hill, 2007.
- [24] S. Paquelet, L. M. Aubert, and B. Uguen, "An impulse radio asynchronous transceiver for high data rates," in *2004 International Workshop on Ultra Wideband Systems Joint with Conference on Ultra Wideband Systems and Technologies. Joint UWBST IWUWBS 2004 (IEEE Cat. No.04EX812)*, 2004, pp. 1–5.
- [25] L. Barletta and G. Kramer, "On continuous-time white phase noise channels," in *2014 IEEE International Symposium on Information Theory*, June 2014, pp. 2426–2429.
- [26] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015.
- [27] F. F. Manzillo, A. Clemente, and J. L. Gonzalez-Jiménez, "High-gain D-band Transmitarrays in Standard PCB Technology for Beyond-5G Communications," *IEEE Transactions on Antennas and Propagation*, pp. 1–1, 2019.
- [28] C. Fougstedt, K. Szczerba, and P. Larsson-Edefors, "Low-Power Low-Latency BCH Decoders for Energy-Efficient Optical Interconnects," *Journal of Lightwave Technology*, vol. 35, no. 23, pp. 5201–5207, 2017.
- [29] S. Kestur, J. D. Davis, and O. Williams, "BLAS Comparison on FPGA, CPU and GPU," in *2010 IEEE Computer Society Annual Symposium on VLSI*, 2010, pp. 288–293.
- [30] T. Ben-Nun and T. Hoefler, "Demystifying parallel and distributed deep learning: An in-depth concurrency analysis," *ACM Comput. Surv.*, vol. 52, no. 4, Aug. 2019.
- [31] E. Nurvitadhi, Jaewoong Sim, D. Sheffield, A. Mishra, S. Krishnan, and D. Marr, "Accelerating recurrent neural networks in analytics servers: Comparison of FPGA, CPU, GPU, and ASIC," in *2016 26th International Conference on Field Programmable Logic and Applications (FPL)*, 2016, pp. 1–4.
- [32] E. Nurvitadhi, D. Sheffield, Jaewoong Sim, A. Mishra, G. Venkatesh, and D. Marr, "Accelerating Binarized Neural Networks: Comparison of FPGA, CPU, GPU, and ASIC," in *2016 International Conference on Field-Programmable Technology (FPT)*, 2016, pp. 77–84.
- [33] A. Tavanaei, M. Ghodrati, S. R. Kheradpisheh, T. Masquelier, and A. Maida, "Deep learning in spiking neural networks," *Neural Networks*, vol. 111, pp. 47 – 63, 2019.
- [34] J. Kendall, R. D. Pantone, K. Manickavasagam, Y. Bengio, and B. Scellier, "Training end-to-end analog neural networks with equilibrium propagation," *ArXiv*, vol. abs/2006.01981, 2020.
- [35] Agilent technologies, Inc., "Square Law and Linear Detection," Application Note 986, Tech. Rep., 1999.

Towards Implementation of Neural Networks for Non-Coherent Detection MIMO systems

Towards Implementation of Neural Networks for Non-Coherent Detection MIMO systems

Alexis Falempin¹, Julien Schmitt², Trung Dung Nguyen², Jean-Baptiste Doré¹

¹ CEA, Leti, Univ. Grenoble Alpes, F-38000 Grenoble, France

²VSORA, 13 Rue Jeanne Braconnier, 92360 Meudon, France

Email: {alexis.falempin, jean-baptiste.dore}@cea.fr, jschmitt@vsora.com, tdnguyen@vsora.com

Abstract—In this paper, we propose the use of quantized neural networks (NNs) to perform non coherent MIMO detector in sub-TeraHertz (THz) communications. Implementing NNs is challenging because operations are performed using a high number of bits. This results in slow and energy consuming computations. Then, quantization appears to be essential to consider low latency and energy efficient communication systems. Specifically, in this work, we propose quantizing our designed NN performing demapping operation. We employ VSORA's digital signal processor (DSP) architecture to perform the quantization. We observe the impact of quantization on the bit error rate. Moreover, we also evaluate the power computation of the proposed DSP regarding our NN. Our simulation results show that we can quantize the weights of the NN to only 6-bits with neglectable degradation on the performance. Besides, we expect achieving high throughput ($> 1\text{Gbps}$), with a peak power consumption of only 0.58W . Thus, the proposed quantization scheme and DSP design allow to achieve high throughput and high energy efficiency.

Index Terms—Sub-THz communications, MIMO, Neural networks, Quantization, DSP, Energy-efficiency

I. INTRODUCTION

Future wireless communications systems such as 6G systems are in the lead of developing ultra low-latency communications, high data rate applications and improved reliability [1]. In addition, a sustainable design and development of transmitters and receivers components are primordial to enable green communications. To achieve such challenges, new cutting-edge technologies are emerging in wireless communications field such as sub-TeraHertz (THz) communications [2], artificial intelligence (AI) [3] and high efficiency digital signal processing (DSP) units. Such systems may exhibit high complexity and hardware implementation issues especially for AI solutions. Then, in this paper, we demonstrate that quantization of NNs can be used to contribute to the design of low-complexity solutions for sub-THz communications.

Sub-THz communications enable high data rate applications due to the large amount of unused bands [4]. However, sub-THz systems suffer from strong phase impairments due to the inefficiency of oscillators. This issue limits the achievable data rate of these communication systems by causing notable detection errors. To cope with this major issue, a solution is to propose a design of energy receivers coupled to adequate transmissions schemes [5]. In this paper, we use the same system model and scenario description as in [5].

Besides, the breakthrough of machine learning is now anchored in many fields and it is considerably growing in wireless communications field [3][6]. The use of deep learning

solutions allow to solve challenging issues and is then adapted to sub-THz communications. In [5], we developed a NN demapper (NND) which allows to retrieve transmitted symbols within a MIMO channel from simple data and supervised learning. Nonetheless, future communications systems like 6G are in demand of computation efficiency and cost-effective hardware design. Most of the current AI solutions for wireless communications may exhibit issues such as high complexity, hardware implementation and scalability. For instance, most of implementations relying on TensorFlow [7] or PyTorch [8] do not take model optimizations into consideration. Considering quantization, a NN implemented with the previously cited frameworks uses 32-bits to represent floating point numbers, *i.e.* weights, activation functions are represented using 32-bits. This may lead to performance issues regarding memory size, CPU usage and computation speed.

Thus, in this paper, we investigate the use of quantization to optimize the NND proposed in [5]. Specifically, we employ the VSORA Neural Network (VSNN) simulation platform to enable model processing and weights quantization. This platform allows to run cross-compiled applications on a high level model of the DSP which can be configured with various floating point data, quantization patterns and processing power. Floating-point number is represented by its mantissa and exponent [9]. In VSORA ecosystem, quantization is applied on number of bits used for mantissa and exponent independently. In [10], authors conclude that quantization of NNs is currently limited to 8-bits using TensorFlow and PyTorch. VSNN platform allows to lower the number of bits required to represent the NN weights without degrading the performance. Moreover, we instantiate our NND in a simulated DSP to profile the required number of cycles to perform an inference of the NND. This allows to estimate an achievable throughput.

Numerical simulations show that quantization can be lowered to 6-bits for both mantissa and exponent with a slight performance degradation in terms of Bit Error Rate (BER) which can be neglected. Moreover, we can achieve high throughput ($> 1\text{Gbps}$) with a small amount of arithmetic logic units (ALUs) and multiplication plus accumulation (MAC) units.

The remainder of this paper is organized as follows. Sec. II recalls the communication system used in [5]. We introduce the NND in Sec. III including its architecture and training and inference stages. While Sec. IV presents the solution used for quantization and DSP profiling, Sec. V describes the results of

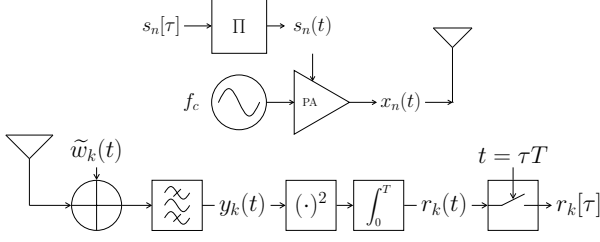


Fig. 1. Block diagram of one Tx-Rx chain

numerical simulations while Sec. VI draws a conclusion with future perspectives.

II. SYSTEM MODEL

In this section, we consider the system model described in [5]. We consider a non coherent MIMO communication system with N_t transmit antennas and N_r receive antennas with $N_t \leq N_r$. Envelope modulation is considered for the transmitter and the receiver is simply an energy detector.

A. Transmitter RF chain

The transmitter implements envelope modulation and the architecture of one of its RF chains is depicted in Fig. 1. On the n -th RF chain, the signal resulting from a rectangular pulse-shaping Π is defined as follow:

$$s_n(t) = \sum_{\tau \in \mathbb{Z}} s_n[\tau] \cdot \frac{\Pi(\frac{t}{T} - \tau - \frac{1}{2})}{\sqrt{T}}, \quad t \in \mathbb{R}, \quad (1)$$

where $s_n[\tau]$ is the τ -th modulated symbol from the On-Off Keying (OOK) constellation $\mathcal{C} = \{0, \sqrt{2}\}$ and T is the symbol duration. We have $\int_{\tau T}^{\tau T + T} |s_n(t)|^2 dt = s_n[\tau]^2$. The transmitted signal $x_n(t)$ at carrier frequency f_c is given by

$$x_n(t) = s_n(t) \cdot \sqrt{2} \cos(2\pi f_c t + \phi(t)), \quad (2)$$

where $\phi(t)$ is a stochastic process modeling a strong oscillator phase noise. The transmitter uses a single oscillator reference, common to all RF chains.

B. Channel model

Sub-THz propagation channels are dominated by a single path, often the LoS direct path, which provides most of the energy contribution [11] [12]. This is due to the usage of directive antennas, sometimes at both transceiver sides. We assume in this work a static LoS channel model.

C. Receiver RF chain

The receiver RF chains architecture is depicted in Fig. 1 for the k -th RF chain. The signal after a band pass filter (with bandwidth $B \geq 2/T$) is given by

$$y_k(t) = \sum_{n=1}^{N_t} h_{k,n} s_n(t) \sqrt{2} \cos(2\pi f_c t + \varphi_{k,n} + \phi(t)) + w_k(t), \quad (3)$$

where $w_k(t)$ is a band-limited continuous Gaussian process with spectral density N_0 , modeling the thermal noise and $h_{k,n}$

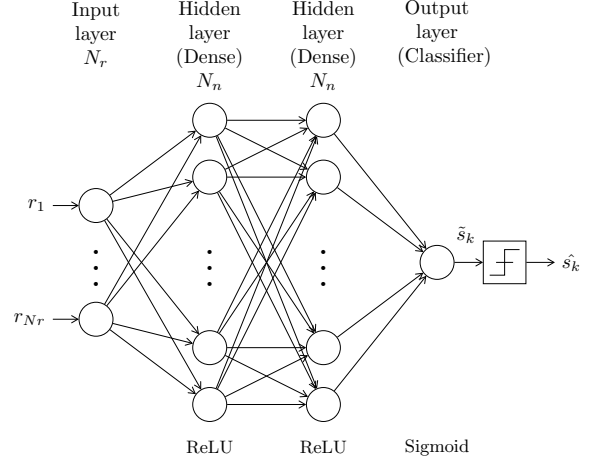


Fig. 2. Architecture of the k -th neural network of the NND

the channel coefficient between TX antenna k and Rx antenna n .

The received symbols are given by [5],

$$r_k[\tau] = E_k[\tau] + \sqrt{2E_k[\tau]} \cdot w_k[\tau] + z_k[\tau], \quad (4)$$

where $w_k[\tau] \sim \mathcal{N}(0, \sigma_w^2)$ is a zero-mean Gaussian variable and $z_k[\tau]$ is a chi-square distributed variable. In case of ideal directive antennas, the co antenna interference is null and the message can be restored. In this work, we assume that interference exists. The complete analysis of such a system has been conducted in [5] and the benefits of considering NN based detectors have been highlighted.

III. NEURAL NETWORK DEMAPPER (NND)

As proposed in [5], using a NN as a demapper allows to propose new perspectives for detection. Indeed, such a solution does not explicitly need the channel propagation matrices to perform symbol estimation. Moreover, it also allows to run symbol detection without any assumption on the channel which is mandatory for Maximum Likelihood Detection under Gaussian Approximation (MLD-GA) algorithm used in previous work.

A. Architecture

The NND is composed of multiple NNs to estimate transmitted symbols. Each of the N_t NN estimates one transmitted symbol. The architecture of the k -th NN is presented on Fig. 2. The NN is composed of N_{hl} fully connected layers. Each layer has N_n neurons and uses Rectified Linear Unit (ReLU) as activation function. The input layer contains N_r entries each one representing a received symbol r_k . Finally, the NN outputs the prediction \tilde{s}_k , homologous to the probability $P_r(s_k = 0|\mathbf{r})$ using a single neuron with sigmoid activation. Besides, each NN can be trained separately, transmitted symbols can be estimated independently.

B. Training

Each NN of the NND is trained independently using reference symbols known at the receiver. Since OOK signalling is used, the NN learns how to retrieve the transmitted symbol

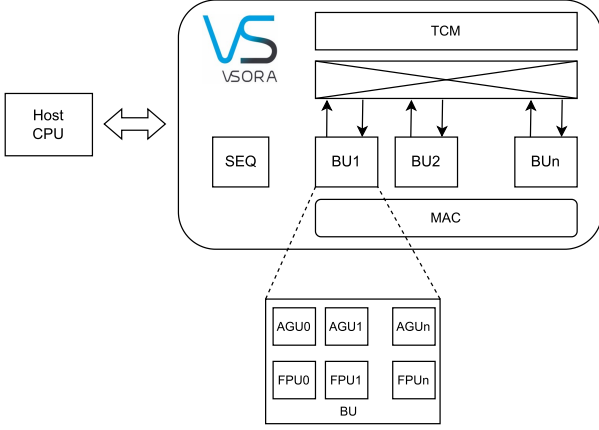


Fig. 3. Architecture of VSORA DSP

using a binary representation of the symbol. Thus, we choose to optimize a Binary Cross-Entropy loss J_k using an Adam optimizer. This loss J_k is described by,

$$-\frac{1}{B_s} \sum_{\tau=1}^{B_s} \left[\frac{s_k[\tau]}{\sqrt{2}} \ln(\tilde{s}_k[\tau]) + \left(1 - \frac{s_k[\tau]}{\sqrt{2}}\right) \ln(1 - \tilde{s}_k[\tau]) \right], \quad (5)$$

where B_s denotes the batch size used during the training.

C. Inference

During inference or online usage, each NN of the NND is considered as a static function applied to the received symbols to retrieve each transmitted symbol s_k . In the scenario presented in Sec. V-A, the wireless link is fixed and the channel may vary very slowly or be even static which does not motivate an online adaptation.

IV. DSP ARCHITECTURE

A. Architecture overview

VSORA's DSP architecture is a single instruction multiple data (SIMD) processor. It is composed of many floating point units (FPUs), all performing the same operation $c = f(a, b)$, where a and b are operands of function f and c is the result, sent by a main sequencer (SEQ) as depicted in Fig. 3. A FPU executes floating point operations. It is associated to an address generator unit (AGU) in charge of generating addresses of operands a , b (read process) and c (write process) which are stored in a tightly coupled memory (TCM).

8 FPUs are gathered in a component called base unit (BU). VSORA's DSP IP is scalable: the number N_{BU} of BUs can be chosen between 1 and 64 (8 – 512 FPUs), targeting a wide range of applications with different capabilities of processing power. Tensor mapping in TCM is optimized in order to feed the FPUs with data each cycle, reaching a rate of use up to 90% for computations such as tensor multiplication. In addition to the FPUs, a block containing MAC operators (Multiplication / Accumulation) has access to all data allowing matrix block multiplication. The number of MAC is configurable using single bloc or multi-blocs and can reach up to $64N_{BU}^2$ MACs per core.

B. Development flow

VSORA neural network development flow (VSNN) is organized around simulation platforms:

- The native platform does not have any representation of the hardware and is used to develop the algorithms and the NN. It is based on standard frameworks (like TensorFlow, ONNX, Caffe ...) associated with a mathematical library developed by VSORA (tensor computation).
- The TLM platform (Transaction Level Model) is a high level representation of the DSP: the model of the DSP is executed on the PC with the application (same code than the native platform) compiled for the target (cross compilation). This platform gives fine estimations of the CPU load and of the memory usage. It is also used for study since it is easy to test a variety of quantization patterns and processing power (number of BUs).
- The register level model (RTL) platform is a low level representation of the DSP and is used for the synthesis of the silicon. It uses the same embedded code as the TLM platform.

C. Data representation

Data are floating point values respecting the IEEE 754 principles, including the denormalization but without specific numbers (infinity, NaN, etc). A word is composed of a mantissa of m bits, an exponent of e bits and a sign bit which defines a quantization $q(e, m)$.

Conventional quantization schemes used in CPU are float (32-bits) (quantization $q(8, 23)$) or double (64-bits) (quantization $q(11, 52)$). VSORA's DSP IP can be configured with a dedicated quantization to optimize the silicon area and the power consumption. With the TLM platform, it is easy to study the impact of various quantizations such as $q(3, 3)$ (7-bits), $q(4, 3)$ (8-bits), $q(3, 4)$ (8-bits), etc without being stuck to a fixed quantization like TensorFlow Lite or PyTorch which is limited to 8 bits. For example, the $q(3, 3)$ pattern characteristics are (i) Lowest denormalized number : 2^{-5} , (ii) Dynamic range : $[-15, 15]$.

D. Compilation process

The compilation process is based on a custom compiler (LLVM based). The system code is implemented in high-level language, using, for example, C++ or Matlab-like / Tensorflow-like code. The code is written as if everything is executed on the host. During compilation, the smart compiler will separate the code running on the VSORA DSP from the code running on the host. For the designer it will look as if everything is executed on the host processor. Using one code for both algorithms and the embedded software makes the design process faster and simpler, which results in a minimal learning curve. For NN codes, a compiler add-on processes the different layers and extracts the corresponding weights.

E. Quantization of Neural Network

VSORA toolchain supports three modes of quantization: Post Training Quantization (PTQ), Light Quantization Aware Training (LQAT) and Quantization Aware Training (QAT).

TABLE I
SIMULATION PARAMETERS

Parameters	Notation	Values
Carrier frequency	f_c	145 GHz
Symbol rate	$1/T$	1 GHz
Bandwidth	$B = 2/T$	2 GHz
Thermal noise	N_0	-174 dBm/Hz
Noise figure	N_f	10 dB
Antenna gain	g_0	32 dBi
Beam width	θ	3 °
Side lobe level	ϵ	-20 dB
Distance Tx - Rx	d_0	10 m

Using PTQ, we directly apply quantization on the pre-trained weights. To reduce accuracy loss, one may need to fine tune the network using a small part of dataset (LQAT mode) or retrain completely with back propagation (QAT mode). PTQ is the fastest but less accurate whereas QAT is the heaviest but most accurate scheme. In this paper, we investigate on PTQ mode.

V. NUMERICAL SIMULATIONS

In this section, we present numerical simulations to evaluate the performance of the proposed quantization for our NN demapper. First, we present the simulation scenario, inspired from [5] which will be used thereafter. Next, we analyse the quantization effect on the NND performance. Eventually, we give an estimation of the achievable throughput using our quantized NN in a simulated DSP unit by the VSORA software.

A. Scenario description

We consider a point to point wireless link in the D-band. Table I describes the simulation parameters. A uniform linear array (ULA) of antennas with $N_t = N_r = N = 4$ is assumed and their specifications are extracted from [13].

Based on the definition of [5], we denote by κ the maximum number of transmitted symbols strongly interfering on a receive antenna. By means of illustration, the channel gain matrix $|\mathbf{H}|$ for $N = 4$ and $\kappa = 5$ may be accurately approximated as follows

$$|\mathbf{H}| \simeq |h_{1,1}| \cdot \begin{bmatrix} 1 & 1 & 1 & \rho \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ \rho & 1 & 1 & 1 \end{bmatrix}, \quad (6)$$

where ρ is the residual interference due to side lobes of the antennas with $\rho^2 = -40$ dB. There are κ diagonals, whose elements are 1, which equals the maximum number of interfering symbols on a receive antenna.

B. Performance assessment

We evaluate the above scenario for a specific case to analyze the proposed quantization and throughput estimation. Thus, here we choose a MIMO system with $N = 4$ and $\kappa = 5$ resulting in strong spatial interference.

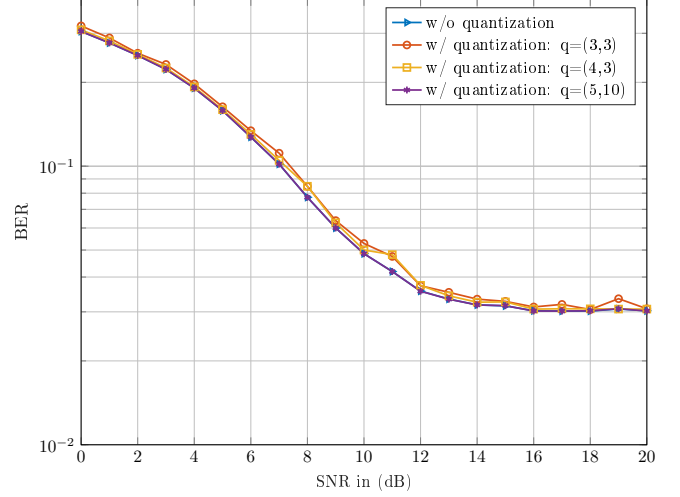


Fig. 4. Impact of quantization on the BER

1) *Post training quantization:* Using VSORA solution and workflow, we perform a post training quantization on the NND, *i.e.* we apply a quantization on the weights learned during the training phase. Several quantization modes are tested. We denote by q the quantization mode used. As stated before in Sec. IV, a floating-point number can be represented using mantissa and exponent bits.

Fig. 4 presents the impact of quantizing NND weights on the Bit Error Rate (BER) performance. First, it must be noted that the BER performance without quantization, referred to as the boundary, is performed using TensorFlow with 32bits floating-point number representation. The BER reaches a an error floor due to the high level of spatial interference. Next, we can observe that using $q = (5, 10)$, we obtain the same performance as boundary. Using $q = (4, 3)$ and $q = (3, 3)$ will induce a slight degradation which can be neglected.

2) *Throughput v.s. Processing power:* In addition to quantization considerations, we perform a throughput estimation using VSORA's DSP. To estimate the achievable throughput, we count the number of cycles required to perform a batched inference of received symbols in function of the DSP resources. The DSP resources are quantified by the number of BUs, *i.e.* the number of MAC operators. In our case, the optimal number of MAC operators, $N_{MAC} = 128N_{BU}$. Besides, one can derive the processing power

$$P_p = 2N_{MAC}F_{clk}, \quad (7)$$

where F_{clk} is the clock frequency of the processor. One MAC is equivalent to 2 Floating Point Operations. Thus, P_p can be expressed in Floating Point Operations per second (FLOPS). Usually, it is expressed in TeraFLOPS (TFLOPS) since high-frequency processors are employed.

In our simulation, we consider a clock frequency $F_{clk} = 2$ GHz of the processor. This allow us to derive the achievable throughput. Fig. 5 presents the theoretical throughput we can achieve regarding the processing power. It must be underlined

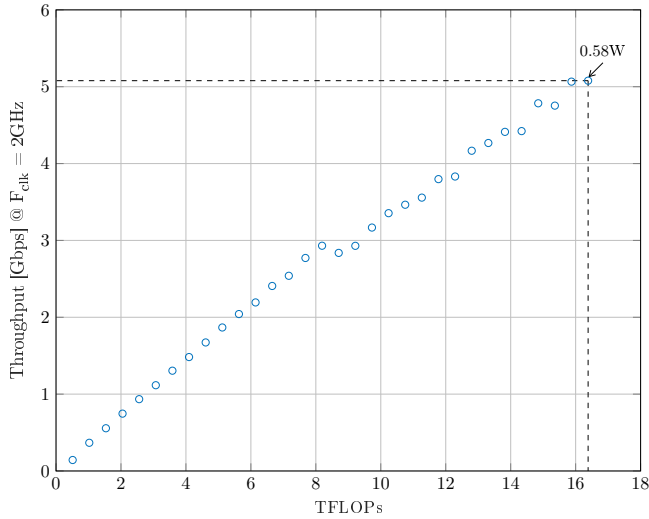


Fig. 5. Achievable throughput in Gbps in function of the processing power

that the provided estimation has been done using a single core for brevity.

Besides, an estimation of the power consumption can be estimated from the simulation environment. If we consider $N_{MAC} = 4096$, leading to ~ 16 TFLOPS, the peak power consumption, *i.e.* when all the hardware resources are used, is approximately 0.58W. This is equivalent to 0.11nJ/bit for a corresponding throughput of 5Gbps.

C. Discussion

In this paragraph, we assess VSORA's DSP against some state-of-the-art solutions in terms of quantization peak power v.s. processing power. First, in terms of quantization, we use a floating-point quantization up to 6-bits whereas other solutions mainly uses int8 quantization [7][14].

Moreover, VSORA's DSP is proposed as an IP and it is possible to choose several parameters: number of computing elements, number of MACs, quantization, TCM memory size. For a dedicated application we can reach an optimum according to the constraints we have to fulfill and since the problem is multidimensional, this optimum could not be trivial. Regarding our proposed application, the main constraint would be maximizing the throughput while optimizing the power consumption. In [14], authors have made a comparison of several solutions in terms of processing power v.s. peak power. The proposed DSP, compared to solutions in the cited work, exhibits the best trade off between processing power and peak power.

Besides, VSORA's development flow offers a complete toolkit to make the best decision and choose the right architecture.

VI. CONCLUSION

In this paper, we studied the use of quantization for a NN performing demapping for sub-THz communications. Quantization is primordial in such systems since low-latency, high throughput and sustainable systems are required. To perform

quantization, we used the VSNN platform to apply PTQ to the NND parameters. Simulation results showed that we can lower the quantization of the weights from 32-bits to only 7-bits with no significant performance loss. Using VSNN, we can achieve a lower quantization than native TensorFlow quantization scheme, limited to 8-bits.

Moreover, using VSORA DSP, we achieve high throughput (> 5 Gbps) for a peak power of only 0.58W leading to 0.116nJ/bit. This demonstrates that our proposed quantization scheme and DSP achieves high data rate with low-energy consumption. Thus, our proposal respects a high performance and low-energy design required in future wireless communications systems.

Finally, future work will involve reducing the quantization while enabling QAT to reduce the impact of the quantization on the performance.

ACKNOWLEDGEMENT

This work has been funded by the H2020-ECSEL CPS4EU European project (grant agreement 826276).

REFERENCES

- [1] E. Calvanese Strinati, S. Barbarossa, J. L. Gonzalez-Jimenez, D. Ktenas, N. Cassiau, L. Maret *et al.*, "6G: The next frontier: From holographic messaging to artificial intelligence using subterahertz and visible light communication," *IEEE Veh. Technol. Mag.*, vol. 14, no. 3, pp. 42–50, Sep. 2019.
- [2] T. S. Rappaport, Y. Xing, O. Kanhere, S. Ju, A. Madanayake, S. Mandal *et al.*, "Wireless communications and applications above 100 ghz: Opportunities and challenges for 6g and beyond," *IEEE Access*, vol. 7, pp. 78 729–78 757, 2019.
- [3] H. Huang, S. Guo, G. Gui, Z. Yang, J. Zhang, H. Sari *et al.*, "Deep Learning for Physical-Layer 5G Wireless Techniques: Opportunities, Challenges and Solutions," *IEEE Wireless Commun.*, vol. 27, no. 1, pp. 214–222, 2020.
- [4] J.-B. Doré, Y. Corre, S. Bicaïs, J. Palicot, E. Faussurier, D. Ktenas *et al.*, "Above-90GHz Spectrum and Single-Carrier Waveform as Enablers for Efficient Tbit/s Wireless Communications," in *25th Int. Conf. on Telecommunications (ICT)*, 2018, pp. 274–278.
- [5] S. Bicaïs, A. Falempin, J.-B. Doré, and V. Savin, "Design and Analysis of MIMO Systems using Energy Detectors for Sub-THz Applications," *IEEE Trans. Wireless Commun.*, pp. 3678–3690, 2021.
- [6] A. Salh, L. Audah, N. S. M. Shah, A. Alhammedi, Q. Abdullah, Y. H. Kim *et al.*, "A Survey on Deep Learning for Ultra-Reliable and Low-Latency Communications Challenges on 6G Wireless Systems," *IEEE Access*, vol. 9, pp. 55 098–55 131, 2021.
- [7] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015.
- [8] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan *et al.*, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," pp. 8024–8035, 2019.
- [9] "IEEE Standard for Floating-Point Arithmetic," *IEEE Std 754-2019 (Revision of IEEE 754-2008)*, pp. 1–84, 2019.
- [10] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, "A Survey of Quantization Methods for Efficient Neural Network Inference," 2021.
- [11] L. Pomietcu and R. D'Errico, "Characterization of Sub-THz and mmWave Propagation Channel for Indoor Scenarios," in *12th European Association on Antennas and Propagation (EurAAP)*, Apr 2018.
- [12] T. Xing and T. S. Rappaport, "Propagation Measurement System and Approach at 140 GHz—Moving to 6G and Above 100 GHz," in *IEEE Global Communications Conf. (GLOBECOM)*, Dec 2018.
- [13] F. F. Manzillo, A. Clemente, and J. L. Gonzalez-Jiménez, "High-gain D-band Transmitarrays in Standard PCB Technology for Beyond-5G Communications," *IEEE Trans. Antennas Propag.*, pp. 587–592, 2019.
- [14] A. Reuther, P. Michaleas, M. Jones, V. Gadepally, S. Samsi, and J. Kepner, "AI Accelerator Survey and Trends," in *2021 IEEE High Performance Extreme Computing Conf. (HPEC)*, 2021, pp. 1–9.

Bibliography

- [1] M. Rahnema, “Overview of the GSM system and protocol architecture,” *IEEE Communications Magazine*, vol. 31, no. 4, pp. 92–100, 1993. (Cited on page 3.)
- [2] 3GPP, “UTRAN overall description,” 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 25.401. (Cited on pages 3 and 78.)
- [3] D. Astely, E. Dahlman, A. Furuskär, Y. Jading, M. Lindström, and S. Parkvall, “LTE: the evolution of mobile broadband,” *IEEE Communications Magazine*, vol. 47, no. 4, pp. 44–51, 2009. (Cited on pages 3 and 78.)
- [4] 3GPP, “Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN): Overall Description,” 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 36.300. (Cited on page 3.)
- [5] ITU, “IMT Vision-Framework and Overall Objectives of the Future Development of IMT for 2020 and Beyond,” International Telecommunication Union (ITU), Recommendation M.2083. (Cited on pages 3 and 78.)
- [6] 3GPP, “Study on Scenarios and Requirements for Next Generation Access Technologies,” 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 38.913. (Cited on pages 3 and 78.)
- [7] E. G. Larsson, O. Edfors, F. Tufvesson, and T. L. Marzetta, “Massive MIMO for next generation wireless systems,” *IEEE Communications Magazine*, vol. 52, no. 2, pp. 186–195, 2014. (Cited on pages 3 and 79.)
- [8] G. Berardinelli, K. Pajukoski, E. Lahetkangas, R. Wichman, O. Tirkkonen, and P. Mogensen, “On the Potential of OFDM Enhancements as 5G Waveforms,” in *2014 IEEE 79th Vehicular Technology Conference (VTC Spring)*, 2014, pp. 1–5. (Cited on pages 3 and 79.)
- [9] R. W. Chang, “Synthesis of band-limited orthogonal signals for multichannel data transmission,” *The Bell System Technical Journal*, vol. 45, no. 10, pp. 1775–1796, 1966. (Cited on page 3.)
- [10] D. Lopez-Perez, A. De Domenico, N. Piovesan, G. Xinli, H. Bao, S. Qitao, and M. Debbah, “A Survey on 5G Radio Access Network Energy Efficiency: Massive MIMO, Lean Carrier Design, Sleep Modes, and Machine Learning,” *IEEE Communications Surveys & Tutorials*, vol. 24, no. 1, pp. 653–697, 2022. (Cited on pages 4 and 79.)
- [11] E. Calvanese Strinati, S. Barbarossa, J. L. Gonzalez-Jimenez, D. Ktenas, N. Cassiau, L. Maret, and C. Dehos, “6G: The next frontier: From holographic messaging to artificial intelligence using subterahertz and visible light communication,” vol. 14, no. 3, pp. 42–50, 2019. (Cited on pages 4 and 79.)
- [12] E. Kolta, T. Hatt, and S. Moore, “Going green: benchmarking the energy efficiency of mobile,” GSMA Intelligence, Tech. Rep., 2021. (Cited on pages 4 and 79.)
- [13] J. von Perner and V. Friderikos, “NETWORK ENERGY EFFICIENCY,” NGMN Alliance, Tech. Rep., 2021. (Cited on page 4.)
- [14] S. Merchan, A. Armada, and J. Garcia, “OFDM performance in amplifier nonlinearity,” *IEEE Transactions on Broadcasting*, vol. 44, no. 1, pp. 106–114, 1998. (Cited on pages 5 and 80.)

- [15] T. Jiang and Y. Wu, "An Overview: Peak-to-Average Power Ratio Reduction Techniques for OFDM Signals," *IEEE Transactions on Broadcasting*, vol. 54, no. 2, pp. 257–268, 2008. (Cited on pages 6, 19 and 80.)
- [16] M. Brandon, M. Ariaudo, S. Traverso, J. Bouvier, I. Fijalkow, and J. L. Gautier, "Linearity improvement thanks to the association of Active Constellation Extension and digital predistortion for OFDM," in *2011 IEEE 9th International New Circuits and systems conference*, 2011, pp. 293–296. (Cited on pages 6 and 81.)
- [17] C. Ni, Y. Ma, and T. Jiang, "A novel constellation amplitude modification method for PAPR reduction in OFDM systems," *Wireless Communications and Mobile Computing*, vol. 16, no. 18, pp. 3307–3315, 2016. (Cited on pages 6 and 81.)
- [18] M. Ben Mabrouk, M. Chafii, Y. Louët, and F. Bader, "Low-PAPR condition for 5G-candidate waveforms," in *2017 XXXIIInd General Assembly and Scientific Symposium of the International Union of Radio Science (URSI GASS)*, 2017, pp. 1–4. (Cited on pages 6 and 81.)
- [19] Y. Roth, J.-B. Doré, L. Ros, and V. Berg, "Coplanar Turbo-FSK: a Flexible and Power Efficient Modulation for the Internet-of-Things," *Wireless Communications and Mobile Computing*, 2018. (Cited on pages 6 and 81.)
- [20] F. Raab, P. Asbeck, S. Cripps, P. Kenington, Z. Popovic, N. Potheary, J. Sevic, and N. Sokal, "Power amplifiers and transmitters for RF and microwave," *IEEE Transactions on Microwave Theory and Techniques*, vol. 50, no. 3, pp. 814–826, 2002. (Cited on pages 6, 19 and 81.)
- [21] H. S. Black, "Inventing the negative feedback amplifier: Six years of persistent search helped the author conceive the idea "in a flash" aboard the old Lackawanna Ferry," *IEEE Spectrum*, vol. 14, no. 12, pp. 55–60, 1977. (Cited on page 6.)
- [22] A. Katz, "Linearization: reducing distortion in power amplifiers," *IEEE Microwave Magazine*, vol. 2, no. 4, pp. 37–49, 2001. (Cited on page 6.)
- [23] A. Kaye, D. George, and M. Eric, "Analysis and Compensation of Bandpass Nonlinearities for Communications," *IEEE Transactions on Communications*, vol. 20, no. 5, pp. 965–972, 1972. (Cited on page 6.)
- [24] C. Jiang, H. Zhang, Y. Ren, Z. Han, K. Chen, and L. Hanzo, "Machine Learning Paradigms for Next-Generation Wireless Networks," *IEEE Wireless Communications*, vol. 24, no. 2, pp. 98–105, April 2017. (Cited on pages 7 and 81.)
- [25] T. O'Shea and J. Hoydis, "[an introduction to deep learning for the physical layer]," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, Dec 2017. (Cited on pages 7 and 81.)
- [26] I. Sohn, "A Low Complexity PAPR Reduction Scheme for OFDM Systems via Neural Networks," *IEEE Communications Letters*, vol. 18, no. 2, pp. 225–228, 2014. (Cited on pages 7 and 81.)
- [27] B. Watkins, R. North, and M. Tummala, "Neural network based adaptive predistortion for the linearization of nonlinear RF amplifiers," in *Proceedings of MILCOM '95*, vol. 1, 1995, pp. 145–149 vol.1. (Cited on pages 7 and 81.)
- [28] S. C. Cripps, *RF Power Amplifiers for Wireless Communications*. Artech, 2006, ch. Power Amplifier Linearization Techniques. (Cited on pages 11, 19 and 81.)
- [29] C. Rapp, "Effects of HPA-nonlinearity on 4-DPSK/OFDM-signal for a digital sound broadcasting system," 10 1991, pp. 179–184. (Cited on pages 14 and 81.)

- [30] Nokia, "Realistic power amplifier model for the New Radio evaluation," 3GPP TSG-RAN WG4, Tech. Rep., 2016. (Cited on pages 14, 64 and 82.)
- [31] A. Saleh, "Frequency-Independent and Frequency-Dependent Nonlinear Models of TWT Amplifiers," *IEEE Transactions on Communications*, vol. 29, no. 11, pp. 1715–1720, 1981. (Cited on pages 14 and 81.)
- [32] J. Staudinger, J.-C. Nanan, and J. Wood, "Memory fading Volterra series model for high power infrastructure amplifiers," in *2010 IEEE Radio and Wireless Symposium (RWS)*, 2010, pp. 184–187. (Cited on pages 14 and 82.)
- [33] P. Gilabert, G. Montoro, and E. Bertran, "On the Wiener and Hammerstein models for power amplifier predistortion," in *2005 Asia-Pacific Microwave Conference Proceedings*, vol. 2, 2005, pp. 4 pp.–. (Cited on pages 15 and 82.)
- [34] W. Huadong, B. Jingfu, W. Zhengde, and H. Jingfu, "An Memory polynomial model for power amplifiers," in *International Conference on Communications, Circuits and Systems*, 2008, pp. 1346–1349. (Cited on pages 15 and 82.)
- [35] R. Zayani, R. Bouallegue, and D. Roviras, "Levenberg-Marquardt learning neural network for adaptive predistortion for time-varying HPA with memory in OFDM systems," in *16th European Signal Processing Conference*, 2008, pp. 1–5. (Cited on pages 15 and 82.)
- [36] 3GPP, "Evolved Universal Terrestrial Radio Access (E-UTRA); Base Station (BS) radio transmission and reception," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 36.106, 2010, release 8. (Cited on page 16.)
- [37] A. D'Andrea, V. Lottici, and R. Reggiannini, "RF power amplifier linearization through amplitude and phase predistortion," vol. 44, no. 11, pp. 1477–1484, 1996. (Cited on page 19.)
- [38] K. Muhonen, M. Kavehrad, and R. Krishnamoorthy, "Look-up table techniques for adaptive digital predistortion: a development and comparison," *IEEE Transactions on Vehicular Technology*, vol. 49, no. 5, pp. 1995–2002, 2000. (Cited on pages 20 and 82.)
- [39] D.-S. Han and T. Hwang, "An adaptive pre-distorter for the compensation of HPA nonlinearity," *IEEE Transactions on Broadcasting*, vol. 46, no. 2, pp. 152–157, 2000. (Cited on page 20.)
- [40] J. Li and J. Ilow, "A least-squares Volterra predistorter for compensation of non-linear effects with memory in OFDM transmitters," in *3rd Annual Communication Networks and Services Research Conference (CNSR'05)*, 2005, pp. 197–202. (Cited on pages 20 and 82.)
- [41] D. Morgan, Z. Ma, J. Kim, M. Zierdt, and J. Pastalan, "A Generalized Memory Polynomial Model for Digital Predistortion of RF Power Amplifiers," *IEEE Transactions on Signal Processing*, vol. 54, no. 10, pp. 3852–3860, 2006. (Cited on pages 21 and 82.)
- [42] M. Schetzen, "Theory of pth-order inverses of nonlinear systems," *IEEE Transactions on Circuits and Systems*, vol. 23, no. 5, pp. 285–291, 1976. (Cited on page 21.)
- [43] R. Zayani, R. Bouallegue, and D. Roviras, "An adaptive neural network pre-distorter for non stationary HPA in OFDM systems," in *15th European Signal Processing Conf.*, 2007, pp. 1352–1356. (Cited on pages 21, 24, 30, 34, 36, 37, 39, 42, 55, 56, 57, 82, 83, 84 and 86.)
- [44] R. Zayani, Y. Medjahdi, H. Bouhadda, H. Shaiek, D. Roviras, and R. Bouallegue, "Adaptive Pre-distortion Techniques for Non-Linearly Amplified FBMC-OQAM Signals," in *IEEE 79th Vehicular Technology Conf. (VTC Spring)*, 2014, pp. 1–5. (Cited on pages 24, 34, 36, 37, 39, 40, 42, 55, 56, 57, 83, 84 and 86.)

- [45] C. Tarver, L. Jiang, A. Sefidi, and J. R. Cavallaro, "Neural Network DPD via Backpropagation through a Neural Network Model of the PA," in *53rd Asilomar Conf. Signals, Systems, and Computers*, 2019, pp. 358–362. (Cited on pages 24, 30, 37, 39, 56, 57, 59, 83, 84 and 86.)
- [46] Y. Wu, U. Gustavsson, A. G. i. Amat, and H. Wymeersch, "Residual Neural Networks for Digital Predistortion," in *IEEE Global Communications Conf. (GLOBECOM)*, 2020, pp. 01–06. (Cited on pages 24, 37, 39, 56, 57, 84 and 86.)
- [47] A. Falempin, R. Zayani, J.-B. Doré, and E. Calvanese Strinati, "Low-Complexity Adaptive Digital Pre-Distortion with Meta-Learning based Neural Networks," in *IEEE 19th Annual Consumer Communications and Networking Conference (CCNC)*, 2022, pp. 453–453. (Cited on page 25.)
- [48] A. Falempin, J.-B. Doré, R. Zayani, and E. Calvanese Strinati, "Low-Complexity Adaptive DPD: From online optimization to meta-learning," *IEEE Transactions on Broadcasting*, 2022. (Cited on pages 25 and 43.)
- [49] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989. (Cited on pages 26 and 28.)
- [50] L. Bottou and O. Bousquet, "The Tradeoffs of Large Scale Learning," in *Advances in Neural Information Processing Systems 20 (NIPS 2007)*, J. Platt, D. Koller, Y. Singer, and S. Roweis, Eds. NIPS Foundation (<http://books.nips.cc>), 2008, pp. 161–168. (Cited on page 27.)
- [51] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Representations by Back-propagating Errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986. (Cited on page 27.)
- [52] J. Bergstra and Y. Bengio, "Random Search for Hyper-Parameter Optimization," *Journal of Machine Learning Research*, vol. 13, p. 281–305, feb 2012. (Cited on page 28.)
- [53] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd Int. Conf. Learning Representations (ICLR)*, 2015. (Cited on pages 32, 34 and 51.)
- [54] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from [tensorflow.org](https://www.tensorflow.org). (Cited on pages 33 and 75.)
- [55] G. Guennebaud, B. Jacob *et al.*, "Eigen v3," <http://eigen.tuxfamily.org>, 2010. (Cited on page 33.)
- [56] Y. Bengio, *Practical Recommendations for Gradient-Based Training of Deep Architectures*. Springer Berlin Heidelberg, 2012, pp. 437–478. (Cited on pages 34, 64 and 88.)
- [57] Xilinx, "Zynq UltraScale+ RFSoc RF Data Converter v2.6 Gen 1/2/3," Tech. Rep., 2021. (Cited on pages 36 and 49.)
- [58] X. Feng, B. Feuvrie, A.-s. Descamps, and Y. Wang, "A digital predistortion method based on nonuniform memory polynomial model using interpolated LUT," in *IEEE Topical Conference on Power Amplifiers for Wireless and Radio Applications (PAWR)*, 2015, pp. 1–3. (Cited on page 42.)
- [59] V.-K. Vu and V.-N. Tran, "Adaptive Digital Predistortion of RF Power Amplifiers Based on Memory Polynomial Model and Indirect Learning Architecture," in *International Conference Engineering and Telecommunication (EnT)*, 2020, pp. 1–5. (Cited on page 42.)

- [60] H. Le Duc, B. Feuvrie, M. Pastore, and Y. Wang, "An Adaptive Cascaded ILA- and DLA-Based Digital Predistorter for Linearizing an RF Power Amplifier," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 3, pp. 1031–1041, 2019. (Cited on page 42.)
- [61] O. Simeone, S. Park, and J. Kang, "From Learning to Meta-Learning: Reduced Training Overhead and Complexity for Communication Systems," in *Proc. 2nd 6G Wireless Summit (6G SUMMIT)*, 2020, pp. 1–5. (Cited on page 42.)
- [62] S. Park, O. Simeone, and J. Kang, "Meta-Learning to Communicate: Fast End-to-End Training for Fading Channels," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 5075–5079. (Cited on pages 42 and 85.)
- [63] S. Park, H. Jang, O. Simeone, and J. Kang, "Learning to Demodulate From Few Pilots via Offline and Online Meta-Learning," vol. 69, pp. 226–239, 2021. (Cited on pages 42 and 85.)
- [64] C. Finn, P. Abbeel, and S. Levine, "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks," in *Proc. 34th Int. Conf. Machine Learning (ICML)*, 2017, pp. 1126–1135. (Cited on pages 43 and 85.)
- [65] R. Z. A. Falempin and J.-B. Doré, "Device for linearizing a power amplifier of a communication system by digital predistortion," French patent US2 107 230, 2022. (Cited on page 43.)
- [66] T. M. Hospedales, A. Antoniou, P. Micaelli, and A. J. Storkey, "Meta-learning in neural networks: A survey," pp. 1–1, 2021. (Cited on pages 43 and 85.)
- [67] M. A. Nielsen, *Neural Networks and Deep Learning*. Determination Press, 2015. (Cited on page 51.)
- [68] X. Zhou, W. ZHANG, Z. Chen, S. DIAO, and T. Zhang, "Efficient Neural Network Training via Forward and Backward Propagation Sparsification," in *Advances in Neural Information Processing Systems*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021. (Cited on page 51.)
- [69] A. Baydin, B. Pearlmutter, A. Radul, and J. Siskind, "Automatic differentiation in machine learning: A survey," *Journal of Machine Learning Research*, vol. 18, pp. 1–43, 04 2018. (Cited on page 51.)
- [70] H. L. Van Trees, *Detection, Estimation and Modulation Theory, Part 1*. Wiley, 1968. (Cited on pages 51 and 86.)
- [71] A. Rajeswaran, C. Finn, S. M. Kakade, and S. Levine, "Meta-Learning with Implicit Gradients," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems (NeurIPS)*, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, Eds., 2019, pp. 113–124. (Cited on page 57.)
- [72] "IEEE Standard for Floating-Point Arithmetic," *IEEE Std 754-2019 (Revision of IEEE 754-2008)*, pp. 1–84, 2019. (Cited on pages 59, 60 and 87.)
- [73] Z. Liu, X. Hu, W. Wang, and F. M. Ghannouchi, "A Joint PAPR Reduction and Digital Predistortion Based on Real-Valued Neural Networks for OFDM Systems," *IEEE Transactions on Broadcasting*, vol. 68, no. 1, pp. 223–231, 2022. (Cited on page 59.)
- [74] A. Fawzy, S. Sun, T. J. Lim, and Y. X. Guo, "An Efficient Deep Neural Network Structure for RF Power Amplifier Linearization," in *IEEE Global Communications Conference (GLOBECOM)*, 2021, pp. 1–6. (Cited on page 59.)
- [75] P. N. Landin, S. Gustafsson, C. Fager, and T. Eriksson, "WebLab: A Web-Based Setup for PA Digital Predistortion and Characterization [Application Notes]," *IEEE Microwave Magazine*, vol. 16, no. 1, pp. 138–140, 2015. (Cited on page 59.)

- [76] M. Horowitz, “1.1 Computing’s energy problem (and what we can do about it),” in *IEEE International Solid-State Circuits Conference (ISSCC)*, 2014, pp. 10–14. (Cited on page 59.)
- [77] M. Nagel, M. Fournarakis, R. A. Amjad, Y. Bondarenko, M. van Baalen, and T. Blankevoort, *A White Paper on Neural Network Quantization*. Chapman and Hall, 2022. (Cited on page 59.)
- [78] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, *A Survey of Quantization Methods for Efficient Neural Network Inference*. Chapman and Hall, 2022. (Cited on page 59.)
- [79] M. Kim, W. Saad, M. Mozaffari, and M. Debbah, “On the Tradeoff between Energy, Precision, and Accuracy in Federated Quantized Neural Networks,” in *International Conference on Communications (ICC)*, 05 2022, pp. 2194–2199. (Cited on page 59.)
- [80] C. Tarver, A. Balatsoukas-Stimming, and J. R. Cavallaro, “Design and Implementation of a Neural Network Based Predistorter for Enhanced Mobile Broadband,” in *IEEE International Workshop on Signal Processing Systems (SiPS)*, 2019, pp. 296–301. (Cited on page 60.)
- [81] A. Falempin, J. Laurent, J.-B. Doré, R. Zayani, and E. C. Strinati, “On the Performance of Quantized Neural Networks based Digital Predistortion for PA linearization in OFDM systems,” in *IEEE Conference on Vehicular Technology (VTC2022-Fall)*, 2022. (Cited on page 60.)
- [82] A. Boutros and V. Betz, “FPGA Architecture: Principles and Progression,” *IEEE Circuits and Systems Magazine*, vol. 21, no. 2, pp. 4–29, 2021. (Cited on page 68.)
- [83] Xilinx, “Zynq UltraScale XZU9EG MPSoC Data Sheet,” Tech. Rep., 2021. (Cited on pages 68 and 69.)
- [84] A. Falempin, J. Schmitt, T. D. Nguyen, and J.-B. Doré, “Towards Implementation of Neural Networks for Non-Coherent Detection MIMO systems,” in *IEEE Conference on Vehicular Technology (VTC2022-Fall)*, 2022. (Cited on page 74.)
- [85] H. Li, Y. Zhang, G. Li, and F. Liu, “Vector Decomposed Long Short-Term Memory Model for Behavioral Modeling and Digital Predistortion for Wideband RF Power Amplifiers,” *IEEE Access*, vol. 8, pp. 63 780–63 789, 2020. (Cited on page 74.)
- [86] R. Zayani, H. Shaïek, and D. Roviras, “Ping-Pong Joint Optimization of PAPR Reduction and HPA Linearization in OFDM Systems,” *IEEE Transactions on Broadcasting*, vol. 65, no. 2, pp. 308–315, 2019. (Cited on page 74.)
- [87] Z. Liu, X. Hu, W. Wang, and F. M. Ghannouchi, “A Joint PAPR Reduction and Digital Predistortion Based on Real-Valued Neural Networks for OFDM Systems,” *IEEE Transactions on Broadcasting*, vol. 68, no. 1, pp. 223–231, 2022. (Cited on page 75.)
- [88] J. Bassey, L. Qian, and X. Li, “A Survey of Complex-Valued Neural Networks,” 2021. (Cited on page 75.)
- [89] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, “Automatic differentiation in pytorch,” in *NIPS-W*, 2017. (Cited on page 75.)
- [90] R. Zayani, H. Shaïek, and D. Roviras, “Efficient Precoding for Massive MIMO Downlink Under PA Nonlinearities,” *IEEE Communications Letters*, vol. 23, no. 9, pp. 1611–1615, 2019. (Cited on page 75.)
- [91] M. Kim, W. Saad, M. Mozaffari, and m. Debbah, “On the Tradeoff between Energy, Precision, and Accuracy in Federated Quantized Neural Networks,” 05 2022, pp. 2194–2199. (Cited on page 76.)

-
- [92] S. Lahmer, A. Khoshsirat, M. Rossi, and A. Zanella, “Energy consumption of neural networks on nvidia edge boards: an empirical model,” 2022. (Cited on page 76.)