Low-Complexity Adaptive DPD for PA linearization: From online optimization to meta-learning

Alexis Falempin, Jean-Baptiste Dore, Rafik Zayani, Emilio Calvanese Strinati

Low-Complexity Adaptive DPD: From online optimization to meta-learning

Alexis Falempin, Jean-Baptiste Doré, Rafik Zayani, Emilio Calvanese Strinati
Univ. Grenoble Alpes, CEA, Leti, F-38000 Grenoble, France
alexis.falempin@cea.fr

***Abstract*—In this paper, we investigate low-complexity and adaptive digital predistorsion (DPD) techniques in order to enhance wireless communication system latency, complexity and power consumption. Specifically, we introduce efficient meta-learning based solutions for time-varying power amplifier (PA) linearization that allow to reduce hardware complexity. Our first proposed solution consists in a meta-learning based neural network (NN) model that is capable to perform, offline, an optimal initialization of the NN based DPD. This leads to an efficient and fast adaptation of the DPD to the time-varying PA characteristics, i.e. few shots are needed during online calibration. Interestingly, we introduced a different approach, referred to as DPD NN Weights Selector (DPDNNWS), that offers the ability to approximate more accurately the current PA characteristic. These solutions have been compared in terms of complexity and performance w.r.t Error Vector Magnitude (EVM) and Adjacent Channel Leakage Ratio (ACLR). Performance of the proposed approaches are benchmarked with the conventional learning based DPD approach. To offer low-complexity DPD, a dedicated NN structure is designed to derive DPD functions using few neurons. Our numerical simulations demonstrated that our proposed DPDNNWS and meta-learning approaches provide satisfying results when used online for several PA models. Contrary to conventional DPD, these approaches can be used to reduce hardware complexity implementation and data usage during online adaptation. Indeed, through numerical evaluation, it appears that meta-learning can reach satisfying performance using only 3000 IQ symbols during online adaptation. Besides, the DPDNNWS approach exhibits performance close to a linear power amplifier. Both solutions allow to achieve excellent performance compared to the state of the art solutions.**

***Index Terms*—OFDM, Energy-efficiency, Power amplifier, Digital pre-distortion, Machine learning, Neural networks, Meta Learning.**

I. Introduction

Modern communication systems such as 5G and Digital Video Broadcasting (DVB) systems are opening the path to low-latency, high data rate communications and connectivity of billions of things. Such improvements lead to key challenges such as energy efficiency and overall system complexity. Both challenges are linked but complexity still represents a major bottleneck concerning hardware design and implementation. Looking at the future, future systems such as 6G [1], a major challenge is to offer a cost-effective service operation and and sustainable design and development of transmitters and receivers hardware components. Specifically, in this paper, we provide solutions to enhance complexity, adaptivity and indirectly energy efficiency of radio-frequency (RF) power amplifiers (PAs). These components are mostly used at transmitter side in order for signals to reach the required transmission distance. Nevertheless, the PA is the most power-hungry component at the RF transmitter. For instance, in 4G systems, the PA consumes about 80% of the total power consumed by the base station [2]. Thus, its power efficiency has to be significantly improved in order to improve the global system energy-efficiency leading to less carbon footprint and energy cost for environmental and economical aspects. Moreover, a RF PA exhibits nonlinear distortions when working close to its saturation level, where its power efficiency is high [3]. Indeed, the PA presents amplitude-to-amplitude (AM/AM) and amplitude-to-phase (AM/PM) distortions, generating severe in-band and out-of-band (OOB) distortions on the wireless link affecting the performance at the receiver. Besides, for DVB and beyond 5G systems, Orthogonal Frequency Division Multiplexing (OFDM) has been widely used because it enables high spectrum efficiency, robustness to frequency selectivity and high data rates [4][5]. Nevertheless, OFDM presents a high Peak to Power Average Ratio (PAPR) which will cause performance degradation because of PA distortions. A common solution is to apply power back-off to OFDM to avoid non linear behavior of the PA. This results in poor energy efficiency. Thus, linearization techniques have been used to cancel non linear effects of PAs while keeping energy efficiency.

Specifically, Digital pre-distortion (DPD) has been widely studied in literature [3], [6] and has been shown to be the most promising PA linearization technique. It consists in adding a module before the PA such that the resulting system DPD and PA is linear. However, estimating the DPD module, *i.e.* estimating the inverse characteristic of the PA is very challenging especially for time-varying PA. DPD has been widely studied and are usually designed with lookup tables [7] or Volterra series [8] but these techniques do not provide adaptive behavior for time-varying PAs. Besides, there has been a growing interest in the use of machine learning techniques for non-linear physical layer design of wireless systems [9] [10]. Indeed, neural networks (NNs) are suited to solve non linear problems. Hence, it is interesting to consider NNs for DPD. This aspect has been widely investigated in the literature. In [11] and [12], authors proposed to use classic neural network architectures using respectively cartesian and polar representation of OFDM signal to perform DPD. Yet, these solutions may require a lot of data to calibrate the model efficiently and are not designed to achieve low-complexity. Similarly, in [13], classic NNs are employed for DPD to

avoid indirect learning algorithm providing satisfying results. In addition, the use of deep NN architectures are also studied in [14] to enable high performance DPD. Although the good performance provided by these solutions, their complexity is still challenging when considering a time-varying PA. Indeed, the updating of the DPD function is performed on a per-PA model, and relearning is needed when the PA behavior changes. Furthermore, the updating process requires sufficient amount of data and training time, which is not adequate with real-time communication systems. To achieve such goals, a new class of training algorithms has emerged which is called meta-learning. Specifically, the use of meta learning in wireless communications is relatively new and few works are related to its usage. In [15], authors claim that meta learning can reduce training overhead and complexity by minimizing pilot symbols usage. Related works [16] and [17] introduce meta learning usage for end-to-end learning and few pilots demodulation. These works show that meta learning significantly reduces data usage and training time. However, to the best of author's knowledge, there is no investigation on the use of meta learning for adaptive DPD.

In order to mitigate the inefficiency in terms of data usage and training time requirements, this work introduces two approaches offering a good adaptation. First, we investigate a meta-learning based NN DPD model which can provide a good initialization of the DPD function offering satisfying performance whatever the PA characteristic. Most importantly, the online calibration of the NN DPD can be efficiently carried out with significant reduced training data and computational complexity contrary to previous works. Second, a classifier based approach relying on generalized likelihood ratio test (GLRT), and a priori known PA characteristics with its corresponding NN DPD weights has been introduced. Its performance is near optimal and requires few data to decide which parameters to use for the DPD function.

The main contributions of this work are threefold. First, we propose a custom architecture specifically dedicated to perform DPD in the polar domain, *i.e.* phase and amplitude domain. Unlike [13],[14] and [11], our architecture tackles separately AM/AM and AM/PM distortions leading to lower complexity and better performance. Second, we propose the study and comparison of two adaptive solutions that relies in different paradigms to provide efficient DPD. A meta-learning approach to provide an adaptive behavior to our NN DPD relying on specific algorithm to improve NN generalization and fast calibration. In this work, we focus essentially on using Model-Agnostic Meta Learning (MAML) algorithm [18]. It is based on finding the best initialization NN weights to converge quickly on later inference. Moreover, we propose a classifier based approach which can select the appropriate NN DPD parameters adopting hypotheses tests between the current PA and a priori known PA characteristics. The latter remains simpler than MAML in terms of implementation and algorithm complexity. Our techniques require few data for online computation which is not the case in [11].

We realize different tests of our solutions on communication system integrating orthogonal frequency division multiplexing (OFDM), quadrature amplitude modulation (QAM) and a PA model derived from 3GPP specification [19].

TABLE I
NOTATIONS

Notations	Meaning
x	lowercase symbol represents a scalar number
$\mathbf{x}$	bold lowercase symbol represents a vector
$\mathbf{X}$	bold uppercase symbol represents a matrix
h	complex perturbation coefficient
N_{fft}	size of FFT for OFDM modulation
N_{CP}	size of cyclic prefix for OFDM
N_{OFDM}	number of OFDM symbols
$\|.\|$	absolute value
$arg(.)$	argument of complex number
p	"knee factor" of Rapp Model, impact on model linearity
ρ	related to DPD for AM/AM distortion
Φ	related to DPD for AM/PM distortion
ω,$\mathbf{W}$, $\mathbf{b}$	trainable parameters of neural networks
$\mathcal{D}$	dataset for neural network training stage
$\mathcal{T}$	task for meta learning
ϕ	inner weights for meta learning
θ	outer weights for meta learning
$\mathcal{L}(\theta, \mathcal{D})$	represent a loss function between dataset $\mathcal{D}$ and predictions of NN using θ
ν	inner learning rate for meta learning
η	outer learning rate for meta learning
∇	gradient operator

The remainder of this paper is structured as follows. Sec. II presents the communication system model. Then, we present our NN DPD in Sec. II-B including NN architecture, training and inference stages. While Sec. III focuses on the study of different approaches to perform NN DPD, Sec. IV presents the simulation results and discussions. Finally, a conclusion is drawn in Sec. V with some future perspectives. For clarity sake, readers may refer to Table I for notations used in this work.

II. System model

A. OFDM transmitter

We consider a communication system integrating a QAM modulation, an OFDM transmitter, a DPD based on NN techniques and a PA. The communication system is pictured on Fig. 1. The loop back link used to derive the DPD function is designed as a complex perturbation coefficient h, *e.g.* phase impairment, and noise $\mathbf{n}$. The signal is then given by,

$$\mathbf{z} = h\mathbf{y} + \mathbf{n}, \tag{1}$$

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.

The PA model is characterized by an amplitude distortion function (AM/AM) denoted f_ρ and a phase distortion function (AM/PM) denoted f_Φ. The output characteristics are given by:

$$|\mathbf{y}| = f_\rho(|\mathbf{x}|) \text{ and } \arg(\mathbf{y}) = f_\Phi(|\mathbf{x}|) + \arg(\mathbf{x}) \tag{2}$$

where $\mathbf{x}$ and $\mathbf{y}$ are the PA input and output signals, respectively. Assuming a PA derived from a 3GPP Rapp model for communication [19], the two functions are defined as follows:

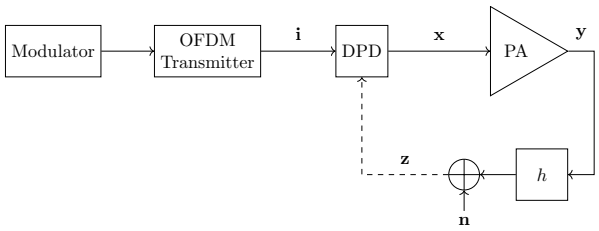
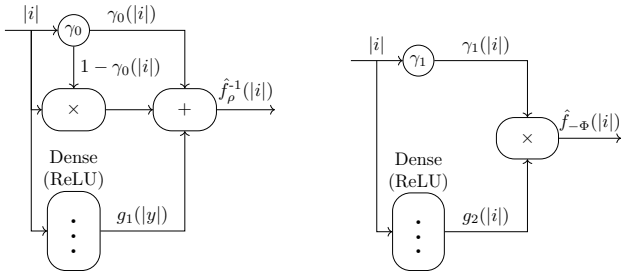


Fig. 1. OFDM system model with a DPD, a PA and a return link



(a) DPD for AM/AM distortion (b) DPD for AM/PM distortion

Fig. 2. Architecture of the LCDPDNN for AM/AM and AM/PM distortions

$$f_\rho(u) = \frac{Gu}{\left(1 + \left|\frac{Gu}{V_{sat}}\right|^{2p}\right)^{\frac{1}{2p}}},\ f_\Phi(u) = \frac{Au^q}{\left(1 + \left(\frac{u}{B}\right)^q\right)} \tag{3}$$

where u denotes the magnitude of the input signal. G represents the gain in linear region, p the "knee" factor and V_{sat} the saturation voltage level. A, B and q are fitting parameters. Thereafter, we consider the following input back-off (IBO) definition :

$$IBO = \frac{P_{sat,in}}{P_{avg,in}}, \tag{4}$$

where $P_{sat,in}$ corresponds to the input power for which the PA reaches its saturation level and $P_{avg,in}$ the average input power.

B. Low-Complexity DPD Neural Network (LCDPDNN)

In this section, we present our solution to perform the NN DPD function. It is based on the indirect learning architecture (ILA), which has been shown to provide better performance compared to the direct learning one. First, it shall be noted that DPD is limited to the transfer function of the power amplifier. To ensure an optimal DPD, the AM/AM distortion must be bijective to estimate its inverse. Second, it is worth to mention that the studied scheme is different compared to the ones presented in the literature [11][13]. Indeed, instead of proposing a "blackbox" architecture, *i.e.* some fully connected hidden layers, we decide to design our neural network to specifically tackle amplitude and phase impairments separately through a polar decomposition of the signal. According to [12], tackling separately the AM/AM and AM/PM distortions gives better results which motivates this choice. It results in a specific architecture.

The LCDPDNN is composed of two neural networks. Each neural network represents a function correcting respectively the amplitude distortion (AM/AM) and the phase distortion (AM/PM). The architecture of both neural networks is presented in Fig. 2. 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 NNs to deal with non linear issues caused by the PA. This allows to enable low-complexity contrary to "blackbox" design.

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 Fig. 2a. 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 Rectified Linear Unit (ReLU) layer is used to improve the estimation produced by the sigmoid. Based on the system model presented in Sec. II, the NN presented here takes the output amplitude of the PA and predicts the input amplitude. This NN is optimized to approximate the function $\widehat{f_\rho^{-1}}$ such as,

$$(f_\rho \circ \widehat{f_\rho^{-1}})(|i|) = G|i|. \tag{5}$$

Looking at Fig. 2a, we define

$$\widehat{f_\rho^{-1}}(u) = g_0(u) + g_1(u), \tag{6}$$
$$\text{where} \begin{cases} g_0(u) = \gamma_0(u) + (1 - \gamma_0(u))\, u, \\ \gamma_0(u) = \left(1 + e^{-\alpha(u-\omega_\rho)}\right)^{-1}, \\ g_1(u) = \sum_{j=1}^{N_n^\rho} \omega_j^\rho \left[\text{ReLU}(\mathbf{W}_\rho(u - \omega_\rho) + \mathbf{b}_\rho)\right]_j, \end{cases}$$

where $\alpha, \omega_\rho, \omega_j^\rho \in \mathbb{R}$, $\mathbf{W}_\rho \in \mathbb{R}^{1\times N_n^\rho}$ and $\mathbf{b}_\rho \in \mathbb{R}^{1\times N_n^\rho}$ are optimized during the learning phase. N_n^ρ denotes the number of neurons. The function g_0 allows to model the invert AM/AM characteristic. Specifically, γ_0 models the correction of the non linearities. The second term of g_0 permits to apply γ_0 after the linear part of the PA. Thus, g_0 allows to correct the non linearities 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. ReLU function is defined by $f(x_i) = \max(0, x_i), x_i = [\mathbf{x}]_i$.

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 invert 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,

$$f_\Phi(|i|) + \hat{f}_{-\Phi}(|i|) = 0. \tag{7}$$

Looking at Fig. 2b, we define

$$\hat{f}_{-\Phi}(u) = \gamma_1(u) g_2(u), \tag{8}$$
$$\text{where} \begin{cases} \gamma_1(u) = \left(1 + e^{-\beta(u-\omega_\Phi)}\right)^{-1}, \\ g_2(u) = \sum_{j=1}^{N_n^\Phi} \omega_j^\Phi \text{ReLU}\left[(\mathbf{W}_\Phi(u-\omega_\Phi) + \mathbf{b}_\Phi)\right]_j, \end{cases}$$

where $\beta, \omega_\Phi, \omega_j^\Phi \in \mathbb{R}$, $\mathbf{W}_\Phi \in \mathbb{R}^{1\times N_n^\Phi}$ and $\mathbf{b}_\Phi \in \mathbb{R}^{1\times N_n^\Phi}$ are optimized during the training phase. N_n^Φ denotes the number of neurons. To design Eq. (8), 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 ReLUs allows to respect the linear zone and correct the non linearites.

III. Digital Pre-Distortion approaches

This paragraph presents three different approaches performing a neural network based DPD to correct distortions induced by PAs.

A. "Conventional" Learning

"Conventional" learning for NNs can be seen as a regular offline training algorithm used for online inference. It is often based on a gradient descend approach to optimize the trainable weights.

In that specific case, our solution is trained using ILA. It consists in deriving a postdistorter and placing it before the PA to perform DPD. This type of approach requires to invert a characteristic and will fully work only with bijective function. 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 DPD and AM/PM DPD NNs.

Then, learning is performed by optimizing a mean squared error (MSE) loss function using the Adam [20] optimizer.

B. Meta-Learning

Meta learning can be seen as a "learn to learn" method meaning that we improve a global learning algorithm using multiple trainings [21]. On the contrary, "conventional" learning improves predictions using a single training on batches of data. On the first hand, conventional learning is said to be task specific, *i.e.* it will learn exclusively for one configuration. On the other hand, meta learning works by learning on multiples configurations called tasks, *i.e.* PA models in our case, such that it can quickly train on a configuration corresponding to the task and adapt a model from few data.

In wireless communication systems, component behaviors can be varying due to many factors like temperature, power level, *etc*. Thus, system configuration often needs to be adapted. For this purpose, meta learning usage can help as well as reducing data cost and training complexity. As stated in the introduction of this work, –readers may refer to Sec. I, meta-learning is promising for wireless communications. Hence, in this section, we consider using meta-learning to perform DPD with few data.

1) Model-Agnostic Meta Learning (MAML): In this subsection, we present the MAML algorithm [18] which is a meta learning algorithm enabling generalization and fast adaptation of neural networks.

The goal of this algorithm is to find the best weights to initialize the neural network for later inference on a dataset. Therein, this algorithm could be seen as an optimal weights initializer. In "conventional" learning, we often randomly initialize neural network weights which can slow the convergence and lead to poor generalization. MAML algorithm can be divided in two major steps, finding the best weights to initialize our NN model and converging faster to the desired weights.

2) Learning weights initialization: This step represents the core of the MAML algorithm. Indeed, the goal of MAML is to quickly train a NN model for a specific configuration. This objective is mainly achieved by finding the weights θ to initialize our NN model. This stage is called meta training.

The weights θ are infered using data from multiple configurations/tasks $\mathcal{T}_i$, belonging to the same distribution $p(\mathcal{T}_i)$. A task $\mathcal{T}_i$ is characterized by two datasets, a training dataset $\mathcal{D}_i^{tr}$ and a test dataset $\mathcal{D}_i^{te}$. More concretely, a task depends on the variation of one or multiple parameters.

As a toy example, we want to derive the function $y = \omega_A \sin(x + \omega_\varphi)$ where ω_A and ω_φ are respectively the amplitude and the phase given in the range Ω_A and Ω_φ. Then, given an unknown ω_A and ω_φ within the range, few samples are required to compute NN weights to retrieve y from x.

MAML algorithm is represented on Fig. 3. The first step consists in sampling a batch of tasks $\mathcal{T}_i$ from $p(\mathcal{T}_i)$. Each task is composed of datasets $\mathcal{D}_i^{tr}$ and $\mathcal{D}_i^{te}$. Then, we derive adapted weights ϕ_i foreach task $\mathcal{T}_i$ using gradient descent. If we consider one gradient step, ϕ_i is given by,

$$\phi_i = \theta - \nu \nabla_\theta \mathcal{L}(\theta, \mathcal{D}_i^{tr}), \tag{9}$$

where ν is the inner learning rate. θ is randomly initialized at the first iteration. Moreover, the loss function $\mathcal{L}$ is evaluated w.r.t. the dataset $\mathcal{D}$ and the prediction made by the NN using parameters θ This step is called meta-update or inner loop and can be seen as a "conventional" learning on each task $\mathcal{T}_i$.

Next, from the optimal weights ϕ_i, MAML infers the weights θ solving the following optimization problem:

$$\min_\theta \sum_i \mathcal{L}(\phi_i, \mathcal{D}_i^{te}). \tag{10}$$

This problem is solved by performing a gradient descent leading to,

$$\theta = \theta - \eta \nabla_\theta \sum_i \mathcal{L}(\phi_i, \mathcal{D}_i^{te}), \tag{11}$$

with η the outer leaning rate. This phase is called the meta-optimization or the outer loop update.

3) Learning adapted weights: Once general weights θ are infered, an adaptation stage must be done to find optimal weights ϕ. This step is straightforward because it represents a "conventional" learning using previously infered θ weights as initialization for our NN. Thus, we get ϕ using gradient descent,

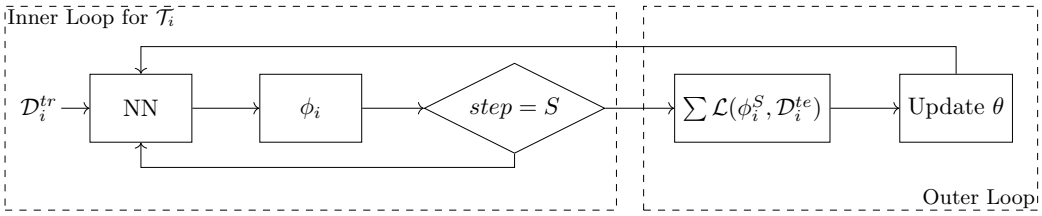


Fig. 3. MAML algorithm state machine

$$\phi^k = \phi^{k-1} - \nabla_{\phi^{k-1}} \mathcal{L}(\phi^{k-1}, \mathcal{D}^{tr}), \tag{12}$$

with $k = 1, \ldots, S$ the number of gradient steps, $\phi^0 = \theta$ and $\mathcal{D}^{tr}$ the samples to update the model to adapted solution.

To fasten this stage, one may consider using an Adam optimizer [20] to find the optimal weights.

4) Improvements to MAML algorithm: First, MAML can be improved to fasten and ensure convergence using loss optimizers instead of regular Stochastic Gradient Descent (SGD) as presented in Eq. (9) and Eq. (11). Some optimizers such as Adam [20] or Rectified Adam [22] greatly improve the convergence by adapting the learning rate of the algorithm.

Then, some core optimizations about the MAML algorithm could be made to improve its convergence. In [23], authors have investigated some improvements that can be done in order to enhance generalization and convergence. Specifically, instead of optimizing the outer loss after doing all the inner steps, it can be envisaged to optimize the outer loss after every inner step. Doing so allows to improve gradient stability of MAML according to the authors.

C. DPD Neural Network Weights Selector (DPDNNWS)

In this paragraph, we present an approach to provide a semi-adaptive DPD solution based on a weights selector. The goal is to select the correct DPD weights to adapt efficiently to a PA change. Indeed, a PA is subject to many environment variations such as the temperature that can modify its characteristic. To provide an efficient DPD function for this scenario, we aim to identify a given PA transfer function based on signal measurements. The DPDNNWS algorithm is then introduced and relies on two major steps.

First, we assume having multiple known PA characteristics denoted PA_k. These characteristics can be acquired using a test bench in a laboratory. The goal of the classifier is to find the closest known characteristic to the current PA characteristic.

Considering the system model described in Sec. II, we assume receiving an unknown signal $\mathbf{z}$, corrupted by a multiplicative complex coefficient and an added noise. The received signal is characterized by an a priori unknown PA characteristic at the transmitter.

In order to decide which class better suits the current PA characteristic, we realize a Generalized Likelihood Ratio Test (GLRT)[24] using the following hypotheses:

$$H_k : \mathbf{z} = h\mathbf{y}_k + \mathbf{n}, \forall k \in E \tag{13}$$

where $\mathbf{y}_k$ denotes the output vector of PA_k and E the set of known PA characteristics. It shall be reminded that $\mathbf{y}_k$ is acquired offline in laboratory. The noise vector $\mathbf{n}$ follows a Gaussian distribution. Thus, the probability density function of z under hypothesis H_k given h and σ^2 is expressed as follows,

$$p_k(\mathbf{z}|h, \sigma^2) = \left(\frac{1}{2\pi\sigma^2}\right)^{N/2} e^{-\frac{1}{2\sigma^2}\|\mathbf{z}-h\mathbf{y}_k\|^2}, \tag{14}$$

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

Then, using Maximum Likelihood Estimation (MLE), we can estimate the quantities h and σ^2 to realize the GLRT. Under hypothesis H_k, $\hat{h}_k$ and $\hat{\sigma_k^2}$, we can show that:

$$\begin{aligned} \hat{h}_k &= \left(\|\mathbf{y}_k\|^2\right)^{-1} \mathbf{z}\mathbf{y}_k^H, \\ \hat{\sigma}_k^2 &= \frac{1}{N}\left\|\mathbf{z} - \hat{h}_k\mathbf{y}_k\right\|^2 \end{aligned} \tag{15}$$

Finally, using the estimators $\hat{h}_k$ and $\hat{\sigma}_k^2$, we can perform the GLRT. By replacing the estimators in Eq. (14), one can define the following minimization rule,

$$k_{opt} = \arg\max_k \left(\|\mathbf{y}_k\|^2\right)^{-1} \left\|\mathbf{z}\mathbf{y}_k^H\right\|^2, \tag{16}$$

with k_{opt} defining the index of the closest PA characteristic.

Second, we suppose that for each known characteristic, we have learned the neural networks weights adapted to fit the DPD using "Conventional" learning described in Sec. III. Thereby, the index k_{opt} allows us to select the best DPD for the current PA.

IV. Numerical simulations

In this section, we present numerical results to underline the benefits of DPDNNWS and MAML approaches compared to "conventional" learning. Thus, we first detail the "conventional" learning performance and limitations in order to promote the other approaches.

We begin considering a time-varying PA model version of the characteristics presented in Sec. II. To demonstrate the benefits of meta learning algorithm for DPD, we compare "conventional" learning and meta learning for adaptation to the parameter p. Thereafter, we consider that $p \in [0.7, 1.5]$. This choice allows to cover a range of PAs with few and lots of non linearities to bring diversity to the simulations. For "conventional" learning, MAML and DPDNNWS algorithms, we consider working on the system model presented in Sec. II with a 64-QAM, and $N_{fft} = 1024$, $N_{CP} = 72$ for OFDM

TABLE II
IMPLEMENTATION PARAMETERS OF LCDPDNN

Parameters	Values
N_n^ρ	6
N_n^ϕ	4
Batch size	128
Total data	2×10^5 IQ symbols
Epochs	50
Optimizer	Adam
Learning rate	0.001
Loss function	Mean Squared Error
Activation function	ReLu

modulation. Besides, we choose a subcarrier spacing of 15kHz. Then, the total bandwidth is 15.36MHz but we only activate 666 subcarriers to reach a 10MHz band for the OFDM signal. This band is frequently used in modern communication systems and is then adapted to the considered scenario. For each algorithm, multiple criteria will be analyzed: Error Vector Magnitude (EVM), spectral regrowth and Adjacent Channel Leakage Ratio (ACLR). Sec. IV-D presents a detailed comparison of each algorithm w.r.t. each metric.

Besides, in this work, we do not take into consideration the memory effects of the PA to better show the efficiency of our proposed approaches for time-varying PAs. Nevertheless, memory effect could be taken into account with multipath effect of the channel at the receiver.

A. Conventional Learning (CL)

In this paragraph, we present the performance of "conventional" learning on static and varying PA model based on Eq. (3). For simulation purpose, we consider that parameters of the PA model are fixed according to [19], *i.e.* $p = 1.1$, $V_{sat} = 1.9V$, $G = 16$, $A = -345$, $B = 0.17$ and $q = 4$.

1) EVM performance: Fig. 4 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. Our solution is very close to the achievable "Limiter" in terms of performance using only 10 neurons which justify the low-complexity aspect. It must be emphasized that such a solution requires a large amount of data for a single state of our PA, about 2×10^5 In-Phase Quadrature (IQ) symbols which represents a cumbersome database for a neural network. The parameters of the LCDPDNN are given in Table II. Besides, we can observe a degradation in high IBO values because we learn on a highly non linear model. Moreover, the architecture is made to cope with non linear models and will be less effective on a PA almost linear.

2) Spectrum analysis: In this paragraph, we introduce spectrum analysis of our DPD. Thereafter, each spectrum 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.

Fig. 5 presents different power spectral densities (PSDs) upsampled according to the left top corner scheme. OFDM

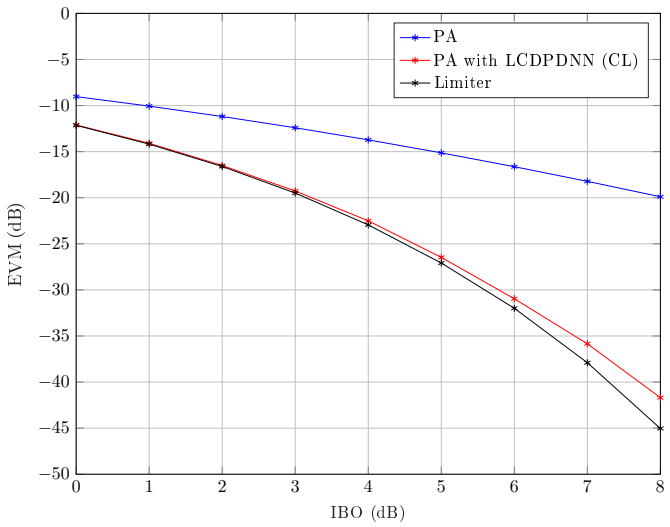


Fig. 4. EVM performance in [dB] using "conventional" learning

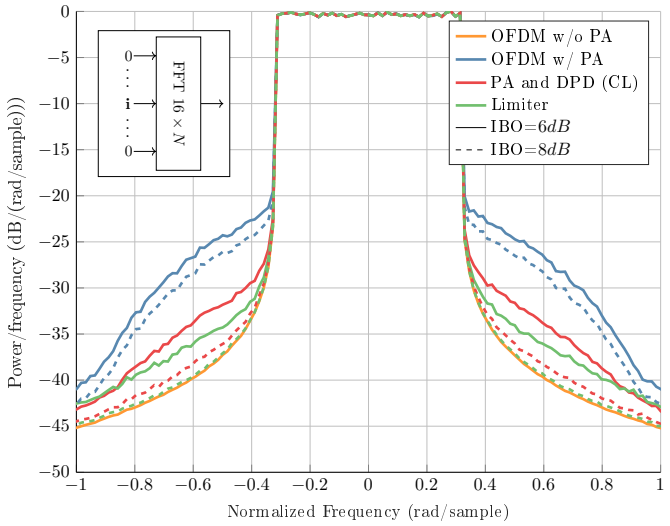


Fig. 5. Spectrum before and after LCDPDNN (CL) using an oversampled FFT, $p = 0.7$

spectrum is pictured in orange and represents our baseline. We can observe that the PA exhibits important spectral regrowth when $IBO = 6$dB or $IBO = 8$dB. In both cases, we see that the use of our DPD allows to achieve great reduction of the spectral regrowth. We almost achieve the same performance as the "Limiter", specially when $IBO = 8$dB.

Second, 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 [25]. This allow us to test our DPD with realistic data converters. A simple scheme is pictured on the top left corner of Fig. 6. Similarly to the Fig. 5, Fig. 6 presents spectrums with $IBO = 6$dB and $IBO = 8$dB. The PA exhibits the same non linearities as presented on Fig. 5. Concerning the DPD, we can state that the spectral regrowth is still considerably reduced. However, using $IBO = 8$dB, we observe that the DPD does not reach

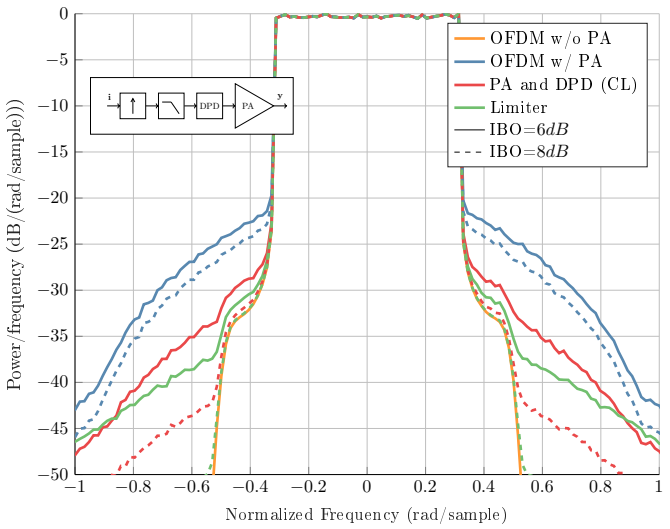


Fig. 6. Spectrum before and after LCDPDNN (CL) with a digital up converter, $p = 0.7$

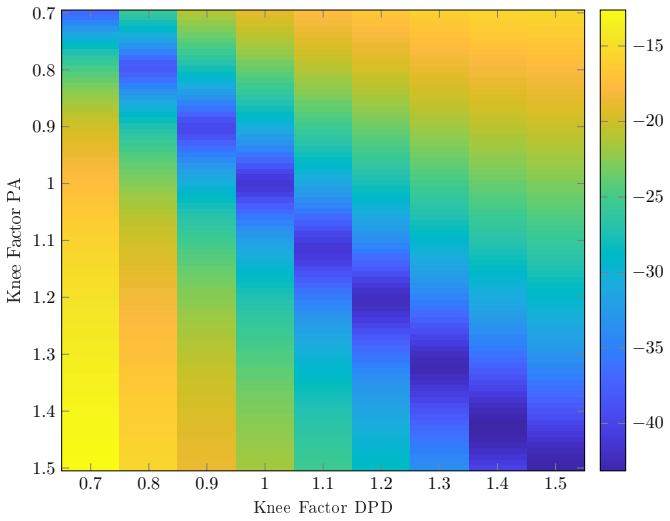


Fig. 7. EVM performance in [dB] with $0.7 \leq p \leq 1.5$, $IBO = 8$dB

the bound of the limiter. At normalized frequency of 0.5, a spectral regrowth appears at -42dB. This slight degradation will not affect the global system performance. Thereafter, the spectrum analysis will be done using the filter presented here.

3) Limitations of conventional approach: To motivate the choice of using different approaches from the conventional one, we present here the limitations of a pretrained DPD using classical training. As a reminder, the goal of our work is to conceive an adaptive DPD solution with low-complexity. We also seek small data usage in order to minimize global latency of the system during online calibration. Thereafter, we consider an adaptive scenario where the parameter p of the PA model Eq. (3) is varying. Fig. 7 presents the EVM performance considering that p is varying with $IBO_{dB} = 8$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 foreach 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 infered p. Moreover, we can underline the fact that the performance is severely degraded when a trained value of p is infered 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. Afterwards, we present the simulation results of our adaptive solutions which bring the benefits of being more flexible and less data consuming than the "conventional" approach.

B. Meta Learning

1) Practical implementation: We give here some detailed practical implementation aspects regarding training and inference of our solution to perform efficient DPD. First, meta learning parameters are summarized in Table III. Next, about the training phase, we consider a task $\mathcal{T}_i$ parametrized by a p value. It must be underlined that the range of tasks must be chosen carefully. Indeed, increasing the value of parameter p will lead to a more linear model of the PA. Hence, having a range of to many linear PAs will decrease the MAML performance for correcting non linearities. Thus, we choose to train our model with $p \in [0.7, 1.5]$. This range ensures to have sufficient non linearities to perform DPD. Moreover, the number of tasks shall also be discussed. In our case, we are trying to minimize the required amount of data to train and infer the algorithm. We have noticed that a total number of 17 tasks is sufficient to have optimal convergence. Increasing this number does not offer a significant gain in terms of performance and will also increase training time.

TABLE III
META LEARNING PARAMETERS

Parameters	Values
Tasks number	17
Inner steps	3
Total data	51000
Epochs	250
Optimizer	Rectified Adam
ν	10^{-3}
η	5×10^{-4}
Loss function $\mathcal{L}$	Mean Squared Error
Activation function	ReLU

Besides, building the database is a crucial step to ensure convergence and performance of the solution. Concerning MAML usage, we must build two datasets, $\mathcal{D}_i^{tr}$ and $\mathcal{D}_i^{te}$ with the same amount of data – refer to Sec. III-B. Both datasets are composed of PA output and input IQ symbols amplitudes. We then have $\mathcal{D}_i^{tr} = (|\mathbf{y}_{tr}|, |\mathbf{x}_{tr}|)$ 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. We consider having for our training $B_{tr} = B_{te} = B_t$. Choosing the value of B_t is important

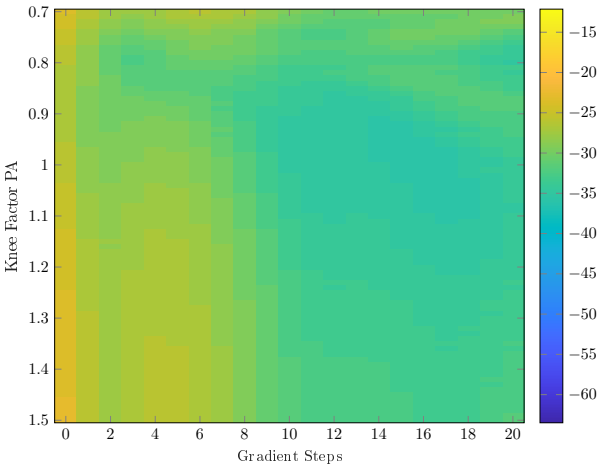


Fig. 8. EVM performance in [dB] with $0.7 \leq p \leq 1.5$, $IBO = 8$dB

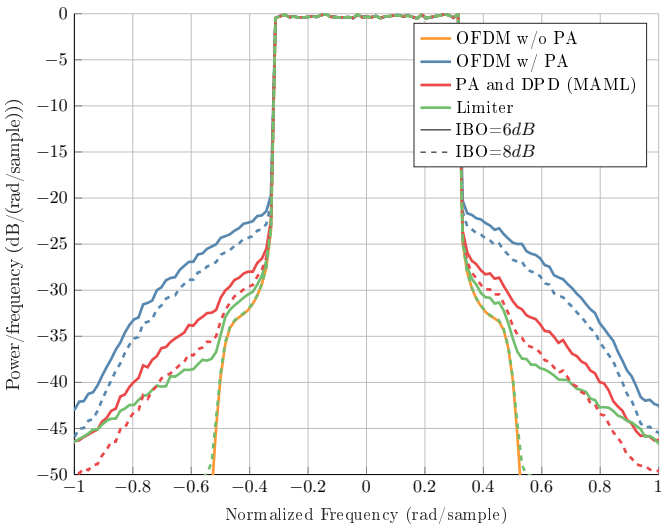


Fig. 9. Spectrum before and after LCDPDNN (MAML), $p = 0.7$

for well generalization of the model. However, choosing a high number of data would significantly impact convergence due to the gradient iteration over multiple tasks. Therefore, we consider having a training batch of IQ symbols with $B_t = 1000k, k \in [1, 10]$. Empirically, we found that using $2000 < B_t < 5000$ gives the best convergence considering that we use multiple inner gradients steps k, *i.e.* the number of inner loop updates in MAML. Thereby, for each inner step k, we perform a gradient descent over 1000 IQ symbols for our batch B_t. Thus, using $B_t = 3000$, we would have 3 inner steps. Using different symbols for each inner step allows to bring more diversity and better convergence. For each task, choosing $B < 1000$ IQ symbols may cause an issue because the statistical distribution of the data will not fully cover the PA model leading to non optimal DPD.

Finally, inference stage is also important. We initialize our LCDPDNN with the learned weights in MAML phase. For any variation of the PA model, an online training will be performed with small gradient updates to improve system latency. In our case, a single batch of 1000 OFDM symbols is used perform to perform online learning for inference.

2) EVM performance: Fig. 8 presents the performance of MAML based DPD over a range of PA with $p = 0.7 + 0.01k, k \in [0, 80]$ in function of gradient steps. First, we can observe that without retraining, *i.e.* gradient steps equals 0, MAML achieves an average of -24dB which is fair compared to conventional learning where we can reach only -10dB in some cases.

Increasing the gradient steps will notably improve the performance. Using 13 gradients steps, we achieve an EVM ranging from -29dB and -35dB regarding the value of p. This is a descent performance considering the wide range of PAs. It shall be reminded that we only used 17 tasks for training. However, here inference is conducted over 80 PA models. It shows that MAML can adapt to unseen tasks but still belonging to the same training task distribution.

3) Spectrum analysis: In this paragraph, we present a spectrum analysis of our LCDPDNN using MAML algorithm for adaptation. Fig. 9 presents the different power density spectrum (PSDs) using the PA only, the LCDPDNN with MAML and a "Limiter" which underlines our performance boundary. Here, we can notice that MAML achieves fair spectrum aliasing correction. Indeed, we can observe that MAML lowers the distortions from the PA. The result is fair considering that we use a generalized model converging in few gradient steps. Regarding the results presented above, we can acknowledge that the use of MAML is pertinent for DPD fast adaptation.

C. DPDNNWS

1) EVM performance: In this paragraph, we evaluate the performance of the weights selector approach presented in Sec. III-C. First, we assume knowing a range of PA characteristics represented by a variation of the p-value. Considering k characteristics, we have $p_k = 0.7 + k\delta_p$, $k \in \mathbb{N} \mid 0 \leq k \leq \frac{1.5-0.7}{\delta_p}$, where δ_p corresponds to a knee factor step. For each p_k, we have a set of corresponding weights that will lead to a near-optimal DPD function.

Fig. 10 presents the performance of the DPDNNWS approach compared to the meta learning one. First, our benchmark evaluates the EVM criterion considering a range of candidate PAs, characterized by a knee factor $p \in [0.7, 0.025, 1.5]$ with an $IBO_{dB} = 8$dB. The blue curve corresponds to the performance of the PA without any DPD function and the green curve corresponds to a linear PA until its saturation.

Then, we underline the performance of the MAML algorithm represented by the pink curve. As we can state, we have a 10dB gain compared to the PA only which is interesting regarding the small amount of updates made to reach this result – see the previous subsection.

Next, we can denote that the DPDNNWS obtains the most satisfying results w.r.t. to the achievable limit. The brown curve corresponds to a weight selector with 9 known PA characteristics, $p_k = 0.7 + 0.1k, k \in \mathbb{N} \mid 0 \leq k \leq 8$. First, we can denote that the selection approach performs better than the meta learning approach even when the DPD is not well classified. Gray squares corresponds to correctly

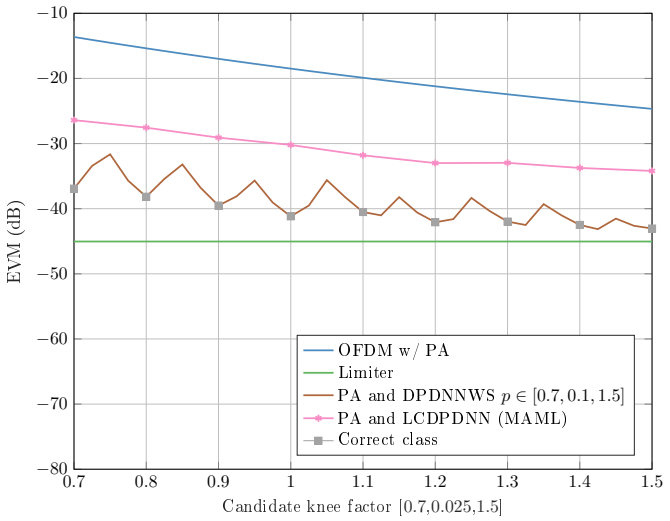


Fig. 10. EVM performance of DPDNNWS and MAML, $IBO = 8$dB

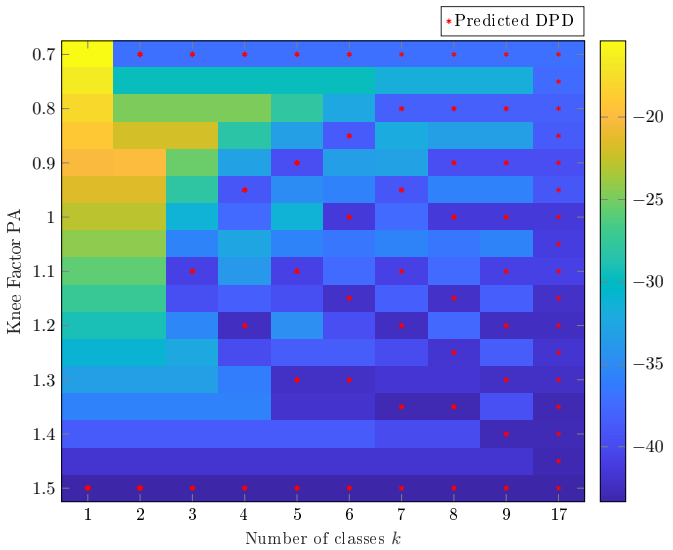


Fig. 11. EVM performance using DPD chosen by the DPDNNWS, $IBO = 8$dB

classified DPD functions. Second, we observe a performance degradation up to 8dB when the DPD is not well chosen by the classifier. Moreover, for accurate weight selection, DPDNNWS needs only 10^3 IQ symbols to perform the DPD choice.

Eventually, Fig. 11 presents the EVM performance considering a DPD chosen by the DPDNNWS. We evaluate the performance of the solution regarding the number of classes. The performance is evaluated considering a PA with $p = 0.7 + 0.05k, k \in \mathbb{N} \mid 0 \leq k \leq 16$.

We consider having a uniformly distributed range of known PA characteristics, *i.e.* classes. The red dots on the figure correspond to the predicted DPD by the classifier. First, it can be noted that performance is severely degraded when the number of known characteristics is small, $k \leq 6$ for low knee factors, *i.e.* more non linear PAs. Above 6 known characteristics, we start having a more homogeneous performance across the range of studied knee factors, $p \in [0.5, 1.5]$. The last column with 17 classes embodies a classifier with knowledge of all characteristics, achieving the best performance with higher complexity. Besides, for this specific algorithm, spectrum analysis is done in Sec. IV-D for more clarity.

D. Discussions

In this paragraph, we compare each algorithm using performance metrics, complexity to better show the relevance of our approaches w.r.t. to classical approaches and state-of-the-art. In the first two paragraphs, we compare our desgined DPDs to the proposed ones in [11] and [12] in terms of performance. Next in the last paragraph, state-of-the-art comparison is provided regarding several criteria.

1) Performance: First, we introduce a spectrum analysis of all the approaches using an $IBO = 8$dB and two tables comparing EVM and ACLR metrics. Fig. 12 presents the PSDs of our developed DPD techniques for $IBO = 8$dB. Second, we can acknowledge that every technique offers a spectral degrowth of the PA distortions. Then, it can be noticed that the use of DPDNNWS permits achieving the same performance as the conventional learning case. Eventually, the use of MAML leads to lessen performance due to its generalization which is under further improvement. All methods allow to decrease significantly spectral regrowth induced by the PA. Besides, from these spectrums, we can derive the ACLR metric. Here, we define it as the ratio between the first left adjacent channel power and the main channel power. Table V presents the ACLR of our techniques. The DPDNNWS is not presented in this table since its performance are the same as the CL case. Using $IBO = 4$dB and $IBO = 6$dB, MAML and DPDNNWS achieve almost the same performance as the linear case with only $2-3$dB difference. When $IBO = 8$dB, we also have an improvement even if our algorithms have a bigger difference with the limiter. This may be due to a fitting issue between the DUC function proposed by [25] and our algorithms which is under investigation.

Besides, we show the performance of DPD used in [11] and [12]. Training of the DPD is respectively performed on cartesian and polar representation of the OFDM symbols. Both solutions employ a fully-connected architecture with ReLUs. We observe an important spectral degrowth compared to the OFDM w/ PA. However, our proposed algorithms still perform better to correct PA non linearities w.r.t. to cited works. ACLR Gain and EVM can be found in Table IV in the last discussion.

2) 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,

$$P_{LCDPDNN} = B_{DPD}(\underbrace{(6N_n^{\rho} + 42)}_{AM/AM} + \underbrace{(6N_n^{\Phi} + 40)}_{AM/PM}), \quad (17)$$

where B_{DPD} denotes the number of symbols during inference stage. This formula takes into account the operations induced by the hidden layers, multiplications, additions and

TABLE IV
COMPARISON WITH STATE-OF-THE-ART

Related Work	ACLR Gain	EVM	FLOPs	Online	Adaptive	IQ symbols for adaptation
RT2DNN [14]	12dB	N/S	412	No	Poor	$\approx 10^6$
DPD w/o ILA [13]	N/S	N/S	184	No	Poor	$\approx 10^4$
Cartesian Dense DPD [11]	9dB	-34dB	322	No	Poor	$\approx 10^6$
Polar Dense DPD [12]	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	$14dB$	$-37dB$	$142 + \mathbf{72}$	Yes	Excellent	$\approx \mathbf{10^3}$

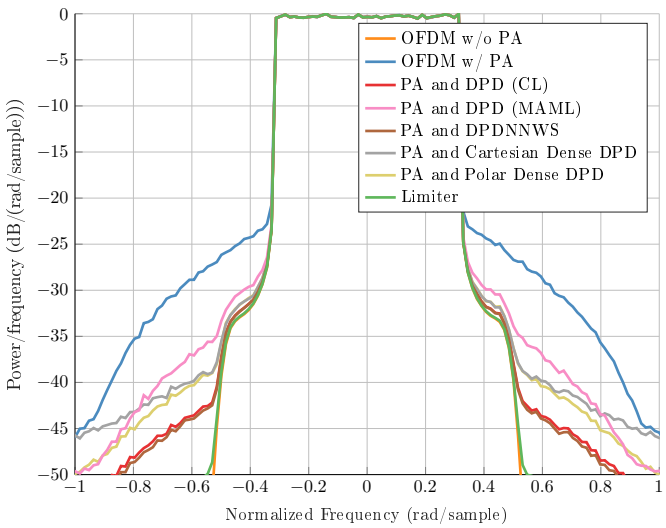


Fig. 12. Spectrum comparison using presented algorithms and solutions from [11] and [12] with $IBO = 8$dB, $p = 0.7$

TABLE V
ACLR (DB)

IBO	PA	Limiter	CL	DPDNNWS	MAML
4dB	-29	-34	-33	-33	-33
6dB	-31	-41	-39	-39	-38
8dB	-33	-53	-47	-47	-41

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 simulation, we use the TensorFlow framework [26] to perform NN training and inference. Computation of exponential relies on the library Eigen [27] 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 5^{th}-order polynomial approximation. Thus, a single sigmoid neuron leads to 27 FLOPs. In addition, it must be underlined that the Eq. (17) does not take into account the complexity of the cartesian to polar conversion, for brevity.

If we consider $B_{DPD} = 1$ and the parameters listed in Table II, $N_n^\rho = 6$ and $N_n^\Phi = 4$, we have $P_{LCDPDNN} = 142$ FLOPs/symbol.

Moreover, regarding the DPDNNWS algorithm, we can also calculate the total number of FLOPs leading to,

$$P_{DPDNNWS} = 8B_{WS}N_E, \tag{18}$$

where B_{WS} denotes the number of symbols required to perform DPDNNWS algorithm and N_E the number of known characteristics. In numerical simulations presented in Sec. IV, $N_E = 9$ leading to 72 FLOPs/symbol.

3) Online usage: Eventually, our proposed solutions are designed to be applied in real-time situations thanks to their high adaptivity. LCDPDNN provides fast predictions during online stage because it consumes a small amount of FLOPs/symbol. However, we cannot consider re-training it for adaptive scenario because it would consume a lot of data and gradient computations. Thus, the use of the proposed MAML and DPDNNWS allow to add this behavior to the proposed DPD. Both adaptive algorithms rely on different paradigms. On the one hand, MAML works by performing an online training of the LCDPDNN using less data and less computation to provide an adapted DPD. On the other hand, DPDNNWS performs a choice of a priori known set of weights to update LCDPDNN based on hypotheses tests. The latter solution needs very few data and FLOPs which makes it the fastest solution for online adaptation. Besides, one can estimate the number of FLOPs required to perform an online calibration with each algorithm. Considering the gradients computations w.r.t. to the parameters and the use of the Adam optimizer, we obtain approximately 1000 FLOPs/symbol/epoch during an online retraining. This leads to 15 TFLOPs. Regarding MAML, we still have 1000 FLOPs/symbol/epoch during an online retraining since we use the LCDPDNN. However, with our proposed approach, we only need 13 epochs and 3000 I/Q symbols leading to 39 MFLOPs. Moreover, with DPDNNWS, we only need 72 kFLOPs. Our results show how MAML and DPDNNWS bring a substantial gain in terms of complexity and are therefore suited for real time usages. In addition, our meta learning algorithm has been tested on single core CPU with a clock frequency of 3.4GHz. This kind of processor can be found in a Software Defined Radio Environment (SDR) – see [28] and references associated for 5G Broadcast. Based on our conducted experiments, the training time has a low probability, 10^{-4}, of being higher than 630 ms and an average time of 619 ms. It is satisfactory for real-time usage. In addition, it should

be mentioned that specialized hardware is currently designed to significantly reduce computational latency.

4) State-of-the-art comparison: In this paragraph, we provide a comparison of our algorithms regarding the state-of-the-art. Table IV presents a comparative analysis of some DPD works and our propositions. Some information cannot be found in cited works and will then be marked as "Not Specified (N/S)". ACLR Gain represents the difference between the PA and the considered DPD technique ACLRs. First, it can be said that our solutions are better in terms of complexity. In related works, the chosen solutions exhibit higher complexity and data usage which prevent an online usage. Besides, MAML and DPDNNWS are efficient for real-time usage because they consume respectively 3×10^3 and 10^3 IQ symbols, *i.e.* 1 to 3 OFDM symbols, to adapt the LCDPDNN weights. Contrary to our solutions, previously cited approaches do not take the adaptive behavior explicitly into account. It results in higher complexity and data usage to provide adapted DPD which is not suited for real-time applications. Second, we compare our solutions to the proposed DPD in [11] and [12]. In those works, it is considered to use only dense architecture on the cartesian or polar representation of the OFDM symbols. Using our simulations parameters, it results in a poorer EVM and ACLR. Those architectures exhibit higher complexity and lesser performance than our model. Besides, in [14], authors proposed a DPD which has excellent performance. However, computational cost is really high resulting in more latency during online usage.

V. Conclusion

In this paper, we proposed a new approach to perform both low-complexity and adaptive DPD based on neural networks, cancelling the non-linear effects caused by PAs in a wireless transmission chain. To achieve these goals, we first designed two custom neural networks that are specifically dedicated to correct respectively AM/AM and AM/PM distortions of PAs. This particular design allows to achieve low-complexity, about 10 neurons for both NNs and only 142 FLOPs which is lower than state-of-the-art approaches. The benefit of this architecture is also to enable fully parallelized operations which drastically reduce computational time compared to classic architecture with many hidden layers. In addition, to answer the adaptive concern of the DPD, we propose the use of two approaches. First, a meta learning approach has been proposed for training and inferring our low-complexity neural networks. Meta learning and specially MAML allows to find the best parameters initialization for the NNs. Based on learning over multiple tasks, we are able to find weights close to the optimal achievable DPD. Next, an online retraining with few samples and few training steps permits achieving fair performance compared to conventional learning. Second, we developed a neural network weights selector based on hypotheses tests with a priori known PA characteristics, and pretrained neural networks DPD weights adapted to known PA characteristics. This solution brings out a real benefit in terms of complexity and data consumption in online mode.

Our numerical results show that meta learning is able to provide functional DPD even for cases unseen during training stage. This implies that this solution is able to adapt easily to a variation of the PA. The DPDNNWS is able to provide even better performance using few known characteristics. Both techniques permit achieving excellent performance with different paradigms. MAML can achieve fast convergence using about 3000 IQ symbols and 13 gradient steps. DPDNNWS achieves optimal weights selection using approximately 1000 IQ symbols and at least 9 known PA characteristics leading to same performance as conventional learning algorithm.

Thus, we truly believe that our proposed solutions bring the key to provide low-complexity, adaptive and efficient DPD for PAs. Further investigation is on going to test our approach on a more realistic system model including a real power amplifier characteristic, analog-to-digital and digital-to-analog converters. Future research direction are on the correction of effects of all these components both in terms of performance and spectral efficiency.

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, 2019.

[2] G. Auer, O. Blume, and V. Giannini, "Energy efficiency analysis of the reference systems, areas of improvements and target breakdown," EARTH, Tech. Rep., 2010.

[3] S. Cripps, "Nonlinear Effects in RF Power Amplifiers," in *RF Power Amplifiers for Wireless Communications, Second Edition*. Artech House, 2006, pp. 231–283.

[4] C. Reiners and H. Rohling, "Multicarrier transmission technique in cellular mobile communications systems," in *Proc. IEEE Vehicular Technology Conf. (VTC)*, vol. 3, 1994, pp. 1645–1649.

[5] W. Zou and Y. Wu, "COFDM: an overview," *IEEE Trans. Broadcast.*, vol. 41, no. 1, pp. 1–8, 1995.

[6] A. D'Andrea, V. Lottici, and R. Reggiannini, "RF power amplifier linearization through amplitude and phase predistortion," *IEEE Trans. Commun.*, vol. 44, no. 11, pp. 1477–1484, 1996.

[7] B. Ai, Z.-x. Yang, C.-y. Pan, S.-g. Tang, and T.-t. Zhang, "Analysis on LUT Based Predistortion Method for HPA with Memory," *IEEE Trans. Broadcast.*, vol. 53, no. 1, pp. 127–131, 2007.

[8] R. N. Braithwaite, "Digital Predistortion of an RF Power Amplifier Using a Reduced Volterra Series Model With a Memory Polynomial Estimator," *IEEE Trans. Microw. Theory Techn.*, vol. 65, no. 10, pp. 3613–3623, 2017.

[9] T. O'Shea and J. Hoydis, "An Introduction to Deep Learning for the Physical Layer," *IEEE Trans. on Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, 2017.

[10] 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.

[11] 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.

[12] R. Zayani, Y. Medjahdi, H. Bouhadda, H. Shaiek, D. Roviras, and R. Bouallegue, "Adaptive Predistortion Techniques for Non-Linearly Amplified FBMC-OQAM Signals," in *IEEE 79th Vehicular Technology Conf. (VTC Spring)*, 2014, pp. 1–5.

[13] 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.

[14] 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.

[15] 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.

[16] 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.
[17] S. Park, H. Jang, O. Simeone, and J. Kang, "Learning to Demodulate From Few Pilots via Offline and Online Meta-Learning," *IEEE Trans. Signal Process.*, vol. 69, pp. 226–239, 2021.
[18] 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.
[19] Nokia, "Realistic power amplifier model for the New Radio evaluation," 3GPP TSG-RAN WG4, Tech. Rep., 2016.
[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd Int. Conf. Learning Representations (ICLR)*, 2015.
[21] T. M. Hospedales, A. Antoniou, P. Micaelli, and A. J. Storkey, "Meta-learning in neural networks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1–1, 2021.
[22] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao *et al.*, "On the Variance of the Adaptive Learning Rate and Beyond," in *Int. Conf. Learning Representations (ICLR)*, 2020.
[23] A. Antoniou, H. Edwards, and A. Storkey, "How to train your MAML," in *Int. Conf. Learning Representations (ICLR)*, 2019.
[24] H. L. Van Trees, *Detection, Estimation and Modulation Theory,Part 1*. Wiley, 1968.
[25] Xilinx, "Zynq UltraScale+ RFSoC RF Data Converter v2.6 Gen 1/2/3," Tech. Rep., 2021.
[26] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org.
[27] G. Guennebaud, B. Jacob *et al.*, "Eigen v3," http://eigen.tuxfamily.org, 2010.
[28] A. Ibanez, J. Sanchez, D. Gomez-Barquero, J. Mika, S. Babel, and K. Kuehnhammer, "5G Broadcast SDR Open Source Platforms," in *IEEE Int. Symp. Broadband Multimedia Systems and Broadcasting (BMSB)*, 2022, pp. 01–06.