IEEE Journal of Microwaves

Received 19 August 2024; revised 18 February 2025 and 2 April 2025; accepted 9 April 2025.
Date of current version 8 May 2025.

Digital Object Identifier 10.1109/JMW.2025.3560420

GPU-Based Implementation of Pruned Artificial Neural Networks for Digital Predistortion Linearization of Wideband Power Amplifiers

WANTAO LI [1], RAÚL CRIADO [1] (Student Member, IEEE), WILLIAM THOMPSON [2], GABRIEL MONTORO [1], KEVIN CHUANG [2] (Senior Member, IEEE), AND PERE L. GILABERT [1] (Senior Member, IEEE)

(Regular Paper)

[1]Department of Signal Theory and Communications, University of Politècnica de Catalunya (UPC) - Barcelona Tech, 08860 Castelldefels, Spain
[2]Aerospace, Defense and Communications Business Unit, Analog Devices Inc. (ADI), Wilmington, MA 01887 USA

CORRESPONDING AUTHOR: Pere L. Gilabert (e-mail: pere.lluis.gilabert@upc.edu).

This work was supported in part by MCIN/AEI/10.13039/501100011033 under Project PID2020-113832RB-C21; in part by MICIU/AEI/10.13039/501100011033/FEDER, UE, under project PID2023-146245OB-C21; in part by the Government of Catalonia; and in part by European Social Fund under Grant 2021-FI-B-137.

ABSTRACT This paper presents a feature selection technique based on ℓ_1 regularization to select the most relevant weights of artificial neural networks (ANNs) for digital predistortion (DPD) linearization of wideband radio-frequency (RF) power amplifiers (PAs). The proposed pruning method is applied to the first hidden layer of a feed-forward real-valued time-delay neural network, commonly used for DPD purposes. In addition, this paper presents the ANN-based DPD implementation using a graphic processing unit (GPU) with compute unified device architecture (CUDA) units. Thanks to the proposed pruning strategy, it is possible to reduce the ANN complexity significantly, thereby achieving a higher data throughput with the GPU-based implementation. The trade-off among RF performance metrics, number of model parameters and throughput of the GPU implementation is evaluated considering the linearization of a high-efficiency pseudo-Doherty load modulated balanced amplifier (LMBA). The linearized PA operating at an RF frequency of 2 GHz delivers a mean output power of 40 dBm with approximately 50% power efficiency when excited with 5G new radio (NR) signals with up to 200 MHz bandwidth and an 8 dB peak-to-average power ratio (PAPR). The real-time GPU implementation of the ANN-based DPD can meet the linearity specifications with a throughput circa 1 GSa/s.

INDEX TERMS Artificial neural network, digital predistortion, graphics processing unit, load-modulated balanced amplifier, model-order reduction.

I. INTRODUCTION

Challenges of beyond 5G (B5G) radio frequency (RF) transmitters include the linearity of high-spectral-efficiency modulated signals and the overall power consumption, i.e., the need to maximize bits/Joule/Hz. The RF power amplifier (PA) is a key component within the wireless transmitter system and, together with baseband signal processing, can significantly impact transmitter power consumption.

Current and B5G orthogonal frequency division multiplexing (OFDM)-based waveforms present significant peak-to-average power ratio (PAPR). Linear amplification of high-PAPR signals requires operating the PA with large power back-off levels, which leads to a serious degradation of average power efficiency. To keep the power efficiency figures high for large back-off levels, highly efficient amplification architectures based on dynamic load modulation (e.g., Doherty PAs [1], [2], [3] or load-modulated balanced amplifiers (LMBA) [4]) or dynamic supply modulation (e.g., envelope tracking (ET) PAs [5]) have been proposed in the literature. Typically, these amplification architectures are designed to

maintain high average efficiency figures not only for large power back-off levels, but also over large bandwidths, i.e., hundreds of MHz. However, when targeting to maximize power efficiency, the linearity is commonly left as a problem to be solved at the system level by using linearizers.

Digital predistortion (DPD) has become one of the most popular linearization techniques, since it is flexible enough to cope with the growing demand of bandwidth, concurrent multi-band and multiple-input multiple-output (MIMO) technologies. DPD can be tailored to linearize different PA architectures, such as ET PAs [6], LMBAs [7], or different technologies, such as concurrent multi-band [8] or MIMO [9], [10] transmissions. In order to characterize and compensate for the unwanted nonlinear distortion and memory effects in PAs, several behavioral models or black-box models have been proposed in the literature [11]. Some of the most commonly used behavioral models are polynomial-based models, such as the memory polynomial (MP) [12] or the generalized MP (GMP) [13], that can be considered as simplified versions of the complex baseband equivalent Volterra series model. Moreover, piece-wise behavioral models, such as the decomposed vector rotation (DVR) model [14], or look-up-table (LUT) implementations [15], [16] are oriented at exploiting the locality of the piece-wise functions. Although these models are nonlinear, they remain linear-in-parameters, making them suitable for extraction using the least squares (LS) solution.

Alternatively, given the outstanding modeling capabilities of artificial neural networks (ANNs), some early works proposed their adoption for RF PA behavioral modeling [17], [18], [19], [20], [21], even at the cost of introducing more complexity to the DPD function than classical polynomial-based behavioral models. More recently, to address the challenge of complex nonlinearity and wideband operating conditions that exhibit strong nonlinear dynamical behavior in highly efficient PAs, ANN-based DPD models [22] have been proposed to overcome the linearization performance limitations of conventional DPD models. However, with ANNs, the linearization performance comes at the price of employing a large number of parameters, which may not be feasible for practical implementations considering both power consumption and resources usage. The ANN-based DPD solution, when considering a vast number of parameters, cannot be implemented for certain clock rates, which ultimately limits the maximum signal bandwidth that can be linearized.

Several publications address the problem of how to derive broadband low-complexity ANN-based DPD models that can provide accurate linearization performance. Most of these works target the complexity reduction by proposing different ANN architectures. For example, a low-complexity augmented real-valued time-delay neural network (ARVTDNN) was presented in [23], a real-valued time-delay convolutional neural network (RVTDCNN) in [24], a mixture of experts neural network in [25], or a phase-normalized neural network (PNN) [26]. Despite these efforts show low-complexity solutions in comparison to other previously published ANN architectures, it is difficult to draw some conclusions since all of them have been tested for different PAs and under different test-signal conditions. In [10] a first attempt to pruning the input layer applying principal component analysis was presented, however the computational complexity reduction achieved without degrading the linearization performance was limited.

In this paper, instead of proposing a specific low-complexity ANN architecture, we propose a generic model-order reduction technique that can reduce the complexity of any ANN models proposed in the literature for linearization purposes. Several feature selection techniques have been proposed in the literature to reduce the complexity of the DPD model and avoid an ill-conditioned estimation [27]. One popular algorithm used to select the most relevant basis functions of the DPD model is the doubly orthogonal matching pursuit (DOMP) [28]. DOMP can be utilized for linear regression DPD models that represent linear combinations of nonlinear basis functions (e.g., GMP and DVR models) and even for N-stage cascaded models [29]. This is because DOMP employs the correlation criterion between residual errors and candidate basis functions, given a reference signal to be modeled. However, the DOMP algorithm cannot be used for ANN-based DPD models, since the ANN is a multi-layer structure, and the reference for different neurons at every layer is unknown.

Therefore, a new pruning approach for ANN DPD models that is based on the ℓ_1 regularization technique is proposed in this paper. This pruning strategy is capable of reducing the number of parameters of the ANN not only without degrading the linearization performance, but also by improving it compared to the original full ANN model (i.e., it shows a regularization effect). Thanks to this general pruning approach, it is possible to scale down the ANN dimensions, which ultimately benefits the hardware implementation of the ANN-based DPD for operating real-time when considering wideband signals.

Digital predistortion linearization is usually computed at baseband using digital signal processing (DSP) units, e.g., a field-programmable gate array (FPGA). Several published works provide details on the DPD implementation in FPGA platforms [30], [31], taking advantages of its reprogrammability, throughput capacity and dedicated DSP resources. The FPGA however, is not the only DSP solution that can offer these advantages. Other architectures, such as the graphic processing unit (GPU), can also provide their own benefits, e.g., large on-device memory and an abundant instruction set. In general, the DPD implementation should evolve with the advance of the manufacturing technology of DSP cores regardless of the platform, i.e., we can always expect that the same DPD algorithm deployed on the latest processors will have better energy efficiency and throughput performance. The implementation of the DPD function in any DSP platform provides a useful baseline (i.e., proof-of-concept) for what will later become an optimal solution, regardless of the platform, such as FPGA, GPU, or application-specific integrated circuit (ASIC).

This paper offers for the first time, to the best of the author's knowledge, details on the GPU implementation of an ANN-based DPD linearizer. Leveraging Nvidia's compute unified device architecture (CUDA) architecture, the GPU implementation is thoroughly explored. To ascertain the real-time viability of the proposed solution, the throughput performance will be assessed across various pruning factors of the ANN-based DPD model. Therefore, the primary contribution of this paper is the GPU-based implementation of a customized ANN for DPD linearization of a wideband, high-efficiency pseudo-Doherty load modulated balanced amplifier (PD-LMBA). This is made possible by the proposed model-order reduction technique, which effectively optimizes ANN complexity and linearization performance while achieving the required throughput of 1 GSa/s.

The remainder of this paper is structured as follows. Section II will initially introduce the phase-normalized neural network employed for DPD within this study. Following that, the iterative training methodology utilized for identifying ANN parameters will be outlined. Lastly, a detailed explanation of the proposed pruning technique will be provided. Moving to Section III, the implementation of GPU-based ANN for DPD will be elaborated upon, alongside a discussion on throughput performance. In Section IV, the experimental setup is described and experimental results showing the linearization performance of the pruned ANN-based DPD are provided. Finally, Section V draws the conclusion.

II. ANN DPD MODEL AND PRUNING TECHNIQUE

In this paper, we adopted the phase-normalized neural network (PNN) DPD model proposed by Fischer-Bühner et al. in [26]. The PNN presents a relatively simple network structure and an effective pre-processing of phase normalization to mitigate the use of complex values. The PNN can outperform other commonly used ANN DPD models, e.g., [22], [24], with fewer parameters.

A. PNN DPD MODEL

Fig. 1 shows the block diagram of the PNN DPD model. It is composed of four parts: (1) feature extraction for calculating the required features for the input layer; (2) input layer that contains all the extracted features; (3) hidden layers of the ANN with, typically, a decreasing number of neurons per hidden layer; (4) output layer containing the real and imaginary part of the phase-normalized output. Finally, the denormalization process is performed to generate the predistortion output sample, $x[n]$.

The ANN input features used in this paper include the phase-normalized input with its delays and the absolute value for several power orders and their respective delays. We denote the phase-normalized input with its delays as a row vector as follows,

$$\boldsymbol{p}_0[n] = (p[n], p[n-1], \ldots, p[n-l] \cdots, p[n-M]), \quad (1)$$

where M is the memory depth and where $p[n-l]$, with $0 \leq l \leq M$, is the phase-normalized input with lagging delay l

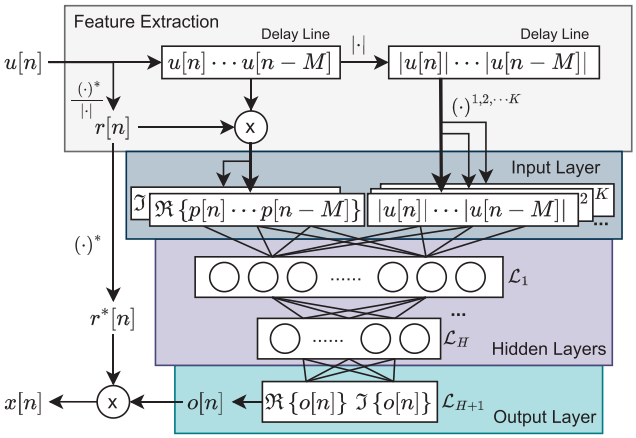


FIGURE 1. Block diagram of the PNN DPD model.

defined as

$$p[n-l] = u[n-l]\, r[n] = u[n-l]\, \frac{u^*[n]}{|u[n]|}, \quad (2)$$

where $u[n]$ is the complex-valued baseband input at discrete time n; and $r[n] = u^*[n]/|u[n]|$ is the normalization factor, where $(\cdot)^*$ denotes complex conjugate. Similarly, we define the following row vectors

$$\boldsymbol{p}_k[n] = \left(|u[n]|^k, \ldots, |u[n-l]|^k, \ldots, |u[n-M]|^k\right) \quad (3)$$

containing the absolute value of $u[n]$ to the power of k and its corresponding delays, where $(1 \leq k \leq K)$, with K being the maximum power order. Finally, the real-valued features for the ANN input layer are defined by concatenation

$$\boldsymbol{p}[n] = \left(\Re\{\boldsymbol{p}_0[n]\}, \Im\{\boldsymbol{p}_0[n]\}, \boldsymbol{p}_1[n], \ldots, \boldsymbol{p}_K[n]\right), \quad (4)$$

with $\Re\{\cdot\}$ and $\Im\{\cdot\}$ denoting the real and imaginary components. Without using the pruning technique, all the neurons are fully connected with the outputs from the previous layer. The output vector of the i^{th} hidden layer is

$$\boldsymbol{o}_{i+1} = \mathcal{L}_i(\boldsymbol{o}_i) = \left(\phi_{i1}(\boldsymbol{o}_i), \phi_{i2}(\boldsymbol{o}_i), \ldots, \phi_{iN_i}(\boldsymbol{o}_i)\right), \quad (5)$$

where $\boldsymbol{o}_i = (o_i[1], o_i[2], \ldots, o_i[N_{i-1}])$ is the output vector from the previous layer, with $\boldsymbol{o}_1 = \boldsymbol{p}[n]$ coming from the input layer, N_i is the number of neurons for the hidden layer $\mathcal{L}_i$, and ϕ_{ij} is the function of the j^{th} neuron at the i^{th} layer, which is defined as a weighted linear function with a bias followed by a nonlinear activation function,

$$\phi_{ij}(\boldsymbol{o}_i) = \tanh\left(\boldsymbol{o}_i \boldsymbol{w}_{ij} + b_{ij}\right), \quad (6)$$

where $\boldsymbol{w}_{ij} = (w_{ij}[1], w_{ij}[2], \ldots, w_{ij}[N_{i-1}])^T$ is a column vector of real-valued parameters (i.e., weights) used for the linear combination (i.e., $\boldsymbol{o}_i \boldsymbol{w}_{ij}$), b_{ij} is a scalar parameter of bias, and $\tanh(\cdot)$ is the hyperbolic tangent. The number of parameters at layer $\mathcal{L}_i$ can be calculated by

$$N_{\mathcal{L}_i} = N_i (N_{i-1} + 1). \quad (7)$$

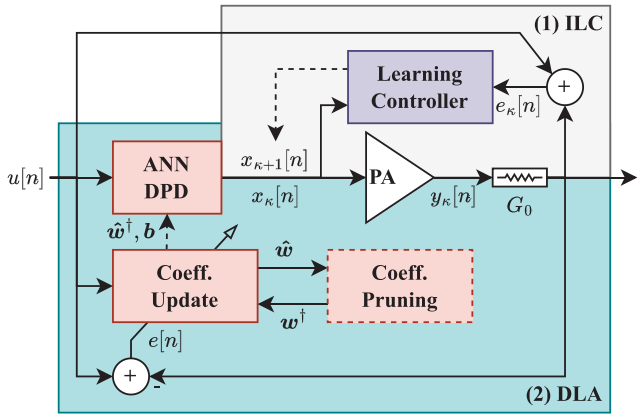


FIGURE 2. Block diagram of the iterative learning control and direct learning approach with parameter pruning.

The output layer is composed of two neurons, the real and imaginary components of the normalized predistorted signal $o[n]$,

$$\boldsymbol{o}_{H+1} = (\Re\{o[n]\}, \Im\{o[n]\}) = \mathcal{L}_{H+1}(\boldsymbol{o}_H), \tag{8}$$

where H is the number of hidden layers. Finally, the predistorted signal is obtained after denomalization

$$x[n] = r^*[n]o[n]. \tag{9}$$

B. ANN-BASED DPD TRAINING

To estimate the parameters of the ANN DPD model fast and efficiently, we consider a two-step procedure. In the first step, we perform the iterative learning control (ILC) process, as described in [32], to find the optimal inverse function of device under test (DUT) as a reference signal. Then, a first estimation or initial condition of the parameters of the ANN DPD model is obtained offline by using the ILC reference signal as the objective output. In the second step, the adaptive online training of the ANN DPD parameters follows a direct learning approach (DLA), as described in [33]. Fig. 2 shows the block diagram of the ILC process required in the first estimation step, the closed-loop DLA required in the second estimation step and the block diagram corresponding to the function for pruning the ANN parameters.

The iterative learning controller directly generates the predistorted signal according to the output error measurement, without using a mathematical model for generating the estimation. Following the notation in Fig. 2, $x_\kappa[n]$ is the complex-valued input signal at iteration κ, while $y_\kappa[n]$ is the measured complex-valued baseband output. The ILC estimation error is computed as

$$e_\kappa[n] = y_\kappa[n]/G_0 - u[n] = e'_\kappa[n] + Z, \tag{10}$$

where G_0 is the expected gain from the PA, e' is the ideal error and $Z \sim N(0, \sigma^2)$ is the inherent measurement noise, with σ being the standard deviation of the test bench noise. The predistorted signal of the next iteration is calculated as follows

$$x_{\kappa+1}[n] = x_\kappa[n] - \lambda_\kappa e_\kappa[n], \tag{11}$$

where λ_κ is the learning rate that can be particularized as described in [34]. Due to the error accumulation, the noise floor increases with every iteration, therefore, the ILC linearization performance stops improving after several iterations. To reduce the noise level, averaging the output signal measurements helps reduce the noise level, but impairments due to additional signal processing, such as capture trigger timing and time-alignment algorithms, may affect the overall system performance.

Denote $\hat{x}_{\text{ILC}}[n]$ the ILC predistorted signal from the last improving iteration that will serve as reference signal for the ANN DPD parameter estimation in the first step. The ANN loss function is the mean square error (MSE) defined as follows

$$E = \frac{1}{2L}\sum_{n=1}^{L} |\hat{x}_{\text{ILC}}[n] - x[n]|^2, \tag{12}$$

with $x[n]$ being the ANN DPD output in (8) and L is the batch size. Finally, in this paper, the ADAM solver from the PyTorch library is used for the ANN parameter identification in this first step.

The accumulated measurement noise in (10) can mask some meaningful distortion signal samples at low power levels and consequently, the ANN model trained taking as the reference signal $\hat{x}_{\text{ILC}}[n]$ may not be optimal. Therefore, in a second step, with the DLA estimation is possible to further improve the linearization performance by capturing the residual error. Let us obviate for now the parameter pruning block in Fig. 2, the DLA runs iteratively and uses the PA output error to improve the linerization performance. Unlike the ILC process, the ANN-based DPD model is now in the loop and (11) is used for generating the objective ANN output. More specifically, $x_{\kappa+1}[n]$ becomes the next ANN learning target while $x_\kappa[n]$ is the ANN output at iteration κ. This second step conforms the closed-loop model training process for the ANN DPD, as described in [33]. The pruning approach for the ANN DPD model is introduced in Section II-C.

C. ANN DPD MODEL PRUNING

The proposed pruning method is based on the ℓ_1 regularization. The solution of least absolute shrinkage and selection operator (LASSO) regression satisfies to the following ℓ_1 optimization problem,

$$\min_{\boldsymbol{w}} \frac{1}{L}\left|\left|\boldsymbol{x}_{\hat{\text{ILC}}} - \boldsymbol{x}\right|\right|_2^2 = \min_{\boldsymbol{w}} \frac{1}{L}\left|\left|\boldsymbol{x}_{\hat{\text{ILC}}} - \text{PNN}(\boldsymbol{u}, \boldsymbol{w})\right|\right|_2^2$$
$$\text{subject to } \left|\left|\boldsymbol{w}\right|\right|_1 \leq t, \tag{13}$$

where L is the signal length, $\boldsymbol{x}$ is the PNN model output, and t is a free parameter that determines the degree of regularization. The goal is to minimize the residual sum of squares subject to a constraint on the sum of the absolute value of the parameters.

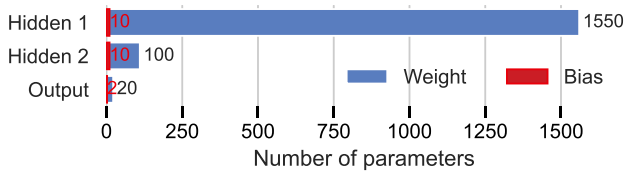


FIGURE 3. Parameter distribution of the PNN model with 2 hidden layers with number of neurons {10, 10}, $M = 30$ and power orders $K = \{1, 3, 5\}$. The model has in total 1692 parameters.

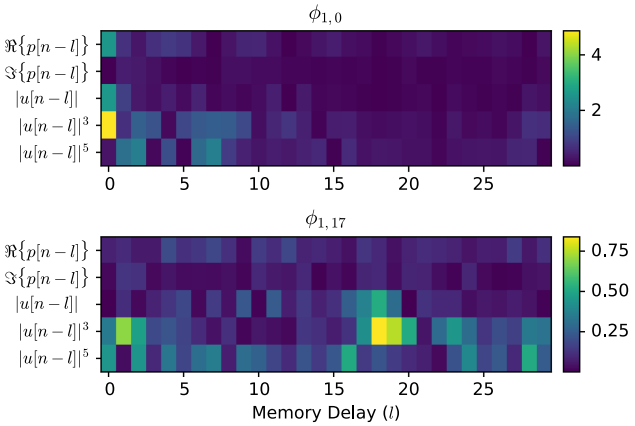


FIGURE 4. Weights of two representative neurons at the first hidden layer, the PNN model has 2 hidden layers with number of neurons {20, 10}.

Before entering into the details of the proposed pruning procedure, we need to understand where to apply the pruning of parameters taking into account the multi-layer ANN architecture. Fig. 3 shows the distribution of parameters of a PNN model with a relatively small configuration, i.e., two hidden layers. As observed, the first hidden layer concentrates most of the parameters (more than 90%). Besides, the multiplicative weights dominate the parameters count, while it is difficult to see the bias in the bar representation in Fig. 3. Therefore, if we can effectively prune the weights from the first hidden layer, the DPD model size will be reduced significantly. Since the first layer connects to the time-delayed features, it can be considered as a 1-dimension convolutional layer or, in other words, a filter. Fig. 4 shows the heat-map of the weights of two neurons (i.e., 0^{th} and 17^{th} neurons) coming from the first hidden layer of a pre-trained PNN model, taking as reference the ILC predistorted signal. The x-axis represents the tap delays and the y-axis the different features (i.e., real and imaginary components of the normalized input and the absolute value of the normalized input to the different power orders), while the color represents the importance (i.e., absolute value) of the weights. As observed, many of the weights present small values but their location cannot be predicted, since the filter parameters are initialized randomly and constructed by the gradient descent.

The pruning of the ANN model is based on ℓ_1 regularization and it is quite straight forward. First, we need to define a percentage threshold ϱ for model reduction. Then, we sort

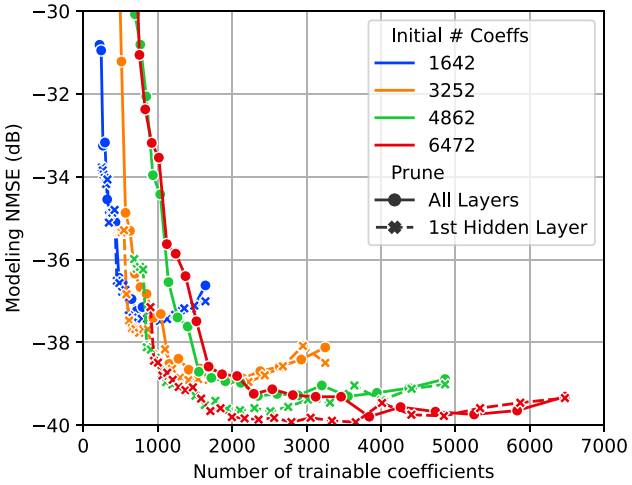


FIGURE 5. Pruning performance in terms of NMSE for modeling the ILC predistorted signal. The PNN model with 2 hidden layers with number of neurons {10|20|30|40, 10}, $M = 30$ and power orders $K = \{1, 3, 5\}$.

the weight parameters from the first hidden layer according to their absolute value. The parameters with absolute value below the threshold ϱ are masked. By applying the mask, we force the selected weight values to 0. During the ANN training process, we set the gradient of the masked weights to zero, so that the masked values are kept unchanged. We could aggressively set $\varrho = 50\%$ to reduce ANN DPD model parameters by half approximately, however, this may remove some useful basis that are eventually important for the modeling performance. To avoid removing relevant weights, we propose to prune the ANN DPD model iteratively with a moderate threshold. Every time, after pruning, the ANN model is allowed to train and update its parameters.

Fig. 5 shows the pruning performance in terms of normalized mean square error (NMSE) between the PNN output in (9) and the reference ILC predistorted signal. As observed in Fig. 5, we tested the PNN pruning considering different test cases with different number of neurons in the first hidden layer and thus, different initial number of parameters. For simplicity, we considered two hidden layers and fixed the number of neurons for the second layer to $N_2 = 10$, while we tested different configurations for the first layer, $N_1 = 10, 20, 30$ and 40, respectively. The model was pruned 30 times with a pruning threshold $\varrho = 10\%$, i.e., at every pruning iteration we remove 10% of the remaining weights. For the identification of the PNN model parameters, we first train the original model, without pruning, for 600 epochs, which is sufficient for the optimizer to converge. Then, after every pruning iteration, we resume the training for another 200 epochs. If no pruning is considered, as observed in Fig. 5, the more initial parameters the PNN has, the better the modeling performance, which is quite intuitive. Let us now focus on the configuration where we prune the first hidden layer of the model with initially 6472 parameters. We can observe that beyond 2000 parameters, the pruning of the parameters does not degrade the modeling performance, on the contrary, it actually improves the NMSE by

Algorithm 1: DLA Parameter Identification With Pruning in The Loop.

1: **procedure** DLA _ Pruning($\boldsymbol{u}, \boldsymbol{x}_{\hat{\text{ILC}}}$)
2: $\boldsymbol{w}_0 \leftarrow$ Random, $\boldsymbol{\varphi} \leftarrow \{\}, \boldsymbol{x}' \leftarrow \boldsymbol{x}_{\hat{\text{ILC}}}$
3: $\hat{\boldsymbol{w}} \leftarrow$ Train_PNN($\boldsymbol{w}_0, \boldsymbol{\varphi}, \boldsymbol{u}, \boldsymbol{x}_{\hat{\text{ILC}}}$) ▷for 600 epochs
4: **for** $c = 1, 2, \ldots, J$ **do**
5: $s \leftarrow |\hat{\boldsymbol{w}}|$
6: $\boldsymbol{w}^\dagger, \boldsymbol{\varphi} \leftarrow$ Prune_PNN($\hat{\boldsymbol{w}}, s, \boldsymbol{\varphi}, \varrho$)
7: **for** $d = 1, 2, \ldots, I$ **do**
8: $\hat{\boldsymbol{w}}^\dagger \leftarrow$ Train_PNN($\boldsymbol{w}^\dagger, \boldsymbol{\varphi}, \boldsymbol{u}, \boldsymbol{x}'$) ▷for 200 epochs
9: $\hat{\boldsymbol{x}} \leftarrow$ Evaluate_PNN($\boldsymbol{u}, \hat{\boldsymbol{w}}^\dagger$)
10: $\boldsymbol{y} \leftarrow$ PA($\hat{\boldsymbol{x}}$)
11: $\boldsymbol{x}' = \hat{\boldsymbol{x}} - \lambda_d(\boldsymbol{y}/G_0 - \boldsymbol{u})$
12: $\boldsymbol{w}^\dagger \leftarrow \hat{\boldsymbol{w}}^\dagger$
13: **end for**
14: $\hat{\boldsymbol{w}} \leftarrow \hat{\boldsymbol{w}}^\dagger$
15: **end for**
16: **return** $\hat{\boldsymbol{w}}^\dagger$
17: **end procedure**

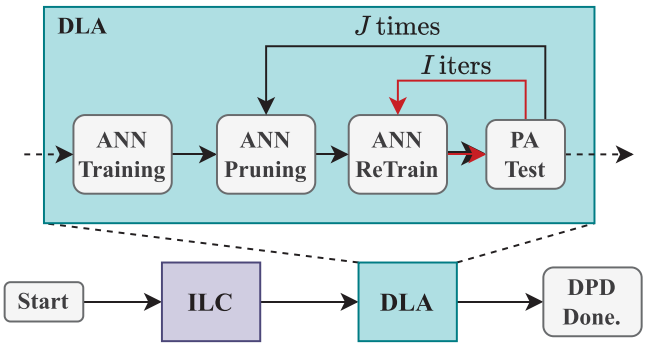


FIGURE 6. Flow chart of the NN DPD combining ILC, DLA and pruning.

around 0.6 dB. With less than 2000 parameters, the modeling performance starts degrading since we are masking weights corresponding to useful features. Taking a look at the NMSE obtained when considering 1500 parameters in Fig. 5, we can observe that the PNN model with initially 6472 parameters can still outperform the PNN model with initially 1642 parameters by more than 2 dB. Besides, by applying pruning to all layers, we lost robustness in comparison to only pruning the parameters of the first hidden layer and thus, the typical NMSE degradation suffered when reducing the number of parameters starts earlier. Therefore, if we have to prune the other layers for an extreme model order reduction, we should first sufficiently prune the first hidden layer before pruning the next layers.

As discussed in Section II-B, the ILC reference signal might not be optimal for training the specific ANN DPD model. Consequently, we propose to include the pruning process in the DLA estimation loop. As depicted in Fig. 2, the pruning operation is included as part of the parameter adaptation process. Fig. 6 shows the flow diagram of the complete ANN DPD process proposed in this paper. It begins with the ILC algorithm for obtaining the DPD reference signal to train the first version of the PNN parameters. During the DLA identification, we prune the pre-trained DPD model for J times. Every time we prune the model, we retrain the PNN model and then evaluate its linearization performance. We repeat the retraining and evaluation for I iterations as part of the regular DLA iterative identification.

Algorithm 1 shows the pseudo code for the pruning process integrated into the DLA parameter identification. Given the baseband input signal $\boldsymbol{u}$ and the predistortion reference signal obtained after the ILC process $\boldsymbol{x}_{\hat{ILC}}$, we pre-train the PNN model for 600 epochs and obtain the parameters $\hat{\boldsymbol{w}}$ for the pre-trained model. The PNN tranining function in line 3 takes as inputs the following arguments: (1) the initial vector of parameters $\boldsymbol{w}_0$ initialized with random values; (2) the masking vector $\boldsymbol{\varphi}$ initialized as an empty set. It stores the indices of the weights being pruned and it is used for the gradient descent optimizer to avoid updating the already pruned parameters; (3) the input signal $\boldsymbol{u}$ to estimate the predistortion output; 4) and the objective output or reference signal $\boldsymbol{x}_{\hat{ILC}}$. Once a pre-trained model is obtained, we enter the outer loop for pruning the PNN DPD model. The pruning function in line 6 of Algorithm 1, takes as input the scores $\boldsymbol{s}$ and the percentage threshold ϱ and updates the indices of the masking vector $\boldsymbol{\varphi}$ accordingly. In addition, it returns the pruned vector of parameters $\boldsymbol{w}^\dagger$, where the less relevant weights taking into account $\boldsymbol{s}$ and ϱ have been set to zero. In the inner loop of regular DLA parameter identification, we allow retraining the PNN parameters for another 200 epochs (line 8) to compensate for the pruning action, as well as for calibrating the model with new measurements. With the pruned and retrained parameters $\hat{\boldsymbol{w}}^\dagger$, the PNN-based DPD generates the predistorted signal $\hat{\boldsymbol{x}}$ (line 9). It is then sent through the PA and, with the new measured output, the objective predistorted signal $\boldsymbol{x}'$ is updated for the next round in line 11. Experimental results of the proposed approach will be shown and discussed in Section IV.

III. GPU-BASED IMPLEMENTATION OF ANN DPD

In this section, we introduce the CUDA GPU-based implementation of an ANN-based DPD optimized for throughput performance. The FPGA is good for processing the signals sample by sample, but it is limited in parallelization due to the hardware resources limitation. Instead, the GPU is more capable of running parallelized threads and processing data by batches. Therefore, parallelizing the DPD algorithm is the fundamental for a GPU-based implementation. Besides, DPD algorithms are highly time correlated, i.e., we need the time delay information of the signal as input to the DPD model. Thus, another key to the GPU-based implementation is to efficiently address the memory resources. Following the CUDA programming guideline [35], threads are grouped into multiple thread blocks, where all the threads in a block run in parallel and multiple blocks can also run in parallel. The maximum number of threads in a block is 1024, and the maximum number of parallel blocks depends on the streaming multiprocessors of the specific GPU device. At the end, the

number of total parallel threads is equal to the number of CUDA cores available in the GPU device. All the CUDA threads can access the GPU device global memory. All threads of a block can share a limited amount of block-level memory resources, which provides higher accessing bandwidth than the global device memory. And finally, each thread has a few local memory for caching the calculation results. In [36] we presented the GPU-based implementation for the classical GMP model. We used one thread to produce one discrete time output and execute as many threads as possible at the same time for parallelization. We used the block shared memory to overcome the bandwidth limitation when multiple threads accessing the global memory interface for acquiring the delayed values. In this paper, we adopt a similar architecture but tailored for the PNN DPD model.

Assuming that the host PC will generate the baseband signal and will send it to the PA device under test (DUT), in order to make the data available for GPU, we need to call the host-to-device (HtoD) memory copy function, and after finishing the calculation, we need to call device-to-host (DtoH) memory copy function to let the output accessible for the host PC to send to the transmitter. Fig. 7 shows the block diagram of the CUDA-based implementation. It graphically exemplifies the concept of multiple threads in a thread block and multiple blocks running in parallel. Every thread is responsible for producing one predistortion output sample, i.e., thread T_1 will produce $x[0]$ and thread T_L will produce $x[L-1]$, in total L threads will be executed for an input batch with L samples. At the beginning, each thread copies the corresponding time-delayed value from the device memory to the shared memory, threads T_1 to T_M need to access the global memory two times for copying and preparing the lagging memories. The function *__syncthreads* is then called to guarantee the coherence of the read back value during calculation. After this preparation, all the threads in the same block start calculating the corresponding PNN output. We focus on showing the data flow of the first thread T_1 in block B_1 for producing the first predistortion output $x[0]$. Since within a thread, there is only series executions and the local memory is very limited, we can avoid caching the results of the input layer by mixing feature extraction with the first hidden layer $\mathcal{L}_1$. Thus, $\mathcal{L}_1$ is composed of a phase-normalized features loop and a power term features loop. Furthermore, according to the distributive law, we can apply the phase-normalization after combining the phase-normalized input features to save multiplication operations. Therefore, the phase-normalized features loop is computed as follows

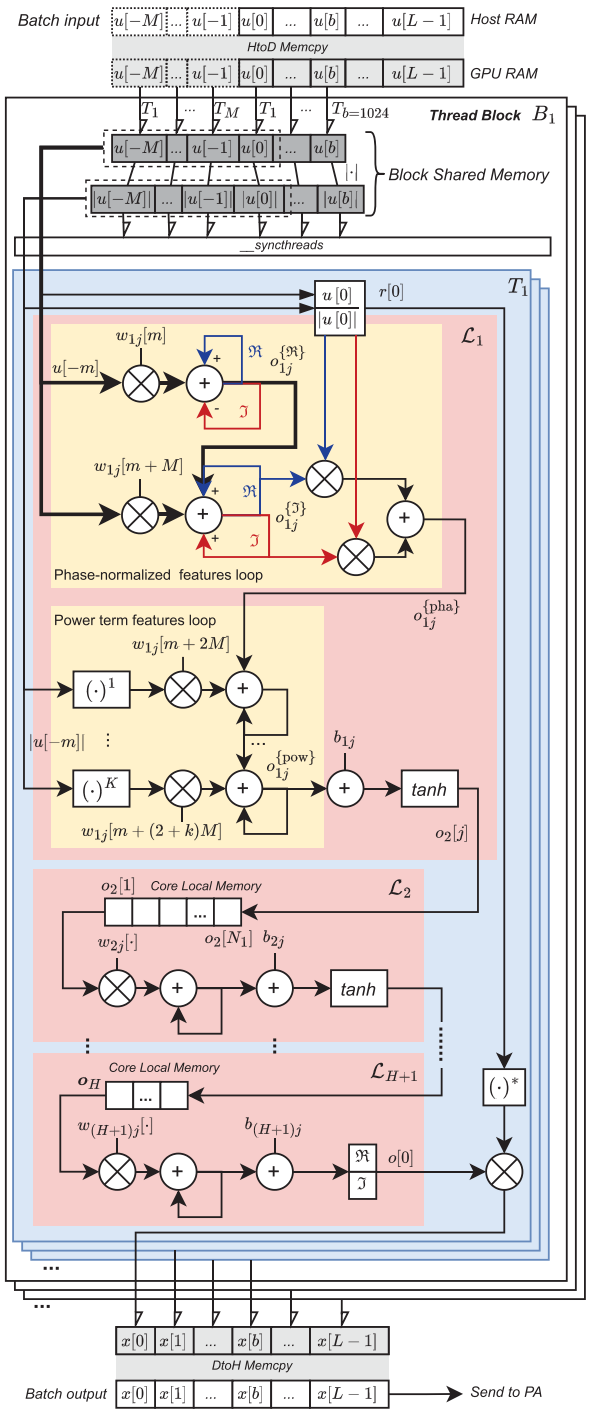


FIGURE 7. Diagram of the CUDA GPU-based PNN DPD implementation.

$$o_{1j}^{\{\Re\}} = \sum_{\forall m \in \mathbb{k}_{j\Re}} w_{1j}[m]\, u[0-m] \tag{14a}$$

$$o_{1j}^{\{\Im\}} = \sum_{\forall m \in \mathbb{k}_{j\Im}} w_{1j}[m+M]\, u[0-m] \tag{14b}$$

$$o_{1\mathrm{j}}^{\{\mathrm{pha}\}} = \Re(r[0])\left(\Re\left(o_{1j}^{\{\Re\}}\right) + \Re\left(o_{1j}^{\{\Im\}}\right)\right) + \Im(r[0])\left(\Im\left(o_{1j}^{\{\Im\}}\right) - \Im\left(o_{1j}^{\{\Re\}}\right)\right) \tag{14c}$$

where $o_{1j}^{\{\Re\}}$ and $o_{1j}^{\{\Im\}}$ stash the accumulation results for the real and imaginary parts, and $o_{1\mathrm{j}}^{\{\mathrm{pha}\}}$ is the result after applying phase-normalization. We define the kernel map $\mathbb{k}$ for storing all the necessary delay values of the pruned model. More specifically, $\mathbb{k}_{j\Re}$ and $\mathbb{k}_{j\Im}$ represent the delays for the real and

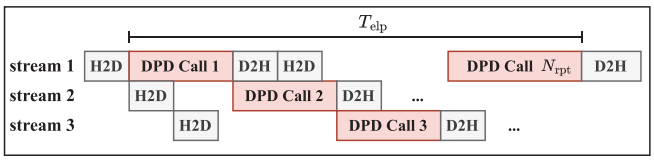


FIGURE 8. Multi-streams concurrent execution to maximize throughtput.

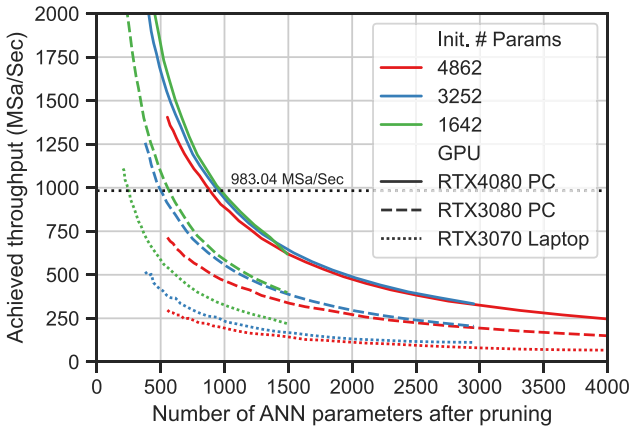


FIGURE 9. Achieved throughput performance of the CUDA-based GPU implementation of the PNN DPD model.

imaginary parts, respectively, of the j^{th} neuron. Similarly, the power term features loop is computed as

$$o_{1j}^{\{\text{pow}\}} = \sum_{k=1}^{K} \sum_{\forall m \in \mathbb{k}_{jk}} w_{1j}\,[m + (2+k)\,M]\,|u\,[-m]|^k \qquad (15)$$

where $\mathbb{k}_{jk}$ contains the delays for the power order k of the j^{th} neuron. Therefore, the output of the neuron j at the first hidden layer is computed by

$$o_2[j] = \tanh\left(o_{1j}^{\{\text{pha}\}} + o_{1j}^{\{\text{pow}\}} + b_{1j}\right) \qquad (16)$$

The computation at the rest of the layers is relatively straight forward by implementing (6) with *for* loops and accumulation. Finally, the output is calculated by $x[0] = o[0]r^*[0]$ with denormalization.

As shown in Fig. 8, we can schedule the execution in multiple concurrent streams, as long as the DPD function execution time is longer than the time for the memory copy, the memory interface will not be the bottleneck for the throughput. Therefore, in this paper, we evaluate the throughput performance of the GPU-based implementation by repeating the DPD function call and taking the average performance, i.e., $(N_{\text{rpt}}L/T_{\text{elp}})$, where N_{rpt} is the number of repeated tests, and T_{elp} is the elapsed time and L is the sample length for each DPD call.

Fig. 9 shows the achieved throughput performance of the proposed implementation. We compared the PNN models pruned with different initial number of parameters, and tested with different GPU devices. We can observe a very predictable

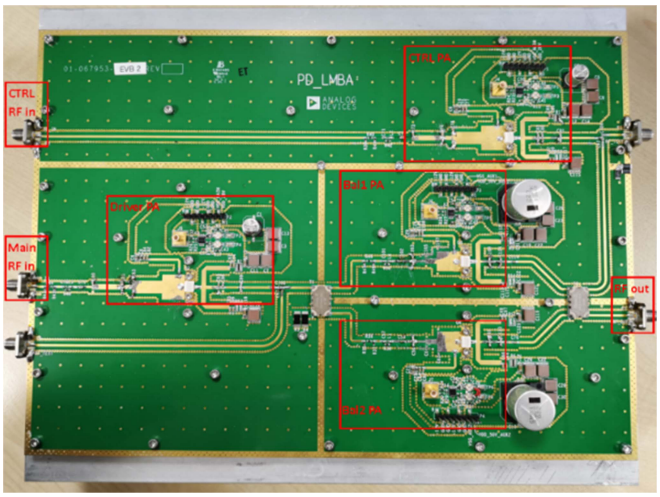


FIGURE 10. A close-up view of the DUT: PD-LMBA evaluation board.

trend, the throughput performance improves when considering more advanced and recent GPUs. With the most recent desktop GPU (i.e., RTX4080), it is possible to achieve around 1 GSa/Sec throughput performance for operating with around 1000 parameters. Therefore, if we consider a transmitter system with a baseband clock of 983.04 MHz, we are now able to implement the PNN DPD model with around 1000 real-valued parameters.

IV. EXPERIMENTAL SET-UP AND RESULTS

A. DEVICE UNDER TEST: PD-LMBA

The LMBA was designed by Analog Devices to be operated as a sequential LMBA [37] as shown in Fig. 10. This configuration is commonly referred to as pseudo-Doherty LMBA (PD-LMBA) [38]. The dual-input PD-LMBA is a topology for high-efficiency operation that comes at the cost of higher intrinsic nonlinearities. The theory of operation is based on the synergy between the Doherty amplifiers and LMBA concepts. Similar to a Doherty, the PD-LMBA consists of the control PA functioning as a carrier PA that is active throughout the entire output range, and the balanced PA acting as the peaking PA that is only active for higher outputs.

The PD-LMBA control stage was constructed using a single 10 W Wolfspeed CGH40010 GaN transistor and designed to operate in class AB. The balanced PA stage utilized two 30 W Wolfspeed CGHV40030 GaN transistors, which were designed to operate in class C. The balanced stage was driven by a single-stage PA constructed using a second 10 W Wolfspeed CGH40010 GaN transistor, which was designed to operate close to class B. Load-pull contours were used to match the devices to the desired gain and efficiency impedances to cover 1 GHz bandwidth centered at 2 GHz. For the input and output 3 dB hybrid, the X3C20A2-03S from Anaren Microwave was used since it covered the design bandwidth and matched to 50 Ohm output. The drain and gate voltage configurations are

TABLE 1. Gate and Drain Configurations of the PD-LMBA

PA Model	Bias descripction	Aprox. quiescent current (mA)	Bias voltage (V)	Drain voltage (V)
Balanced 1	Class-C	0	-3.95	50
Balanced 2	Class-C	0	-3.95	50
Driver	∼ Class-B	5	-3.00	28
Control	Class-AB	50	-2.72	24

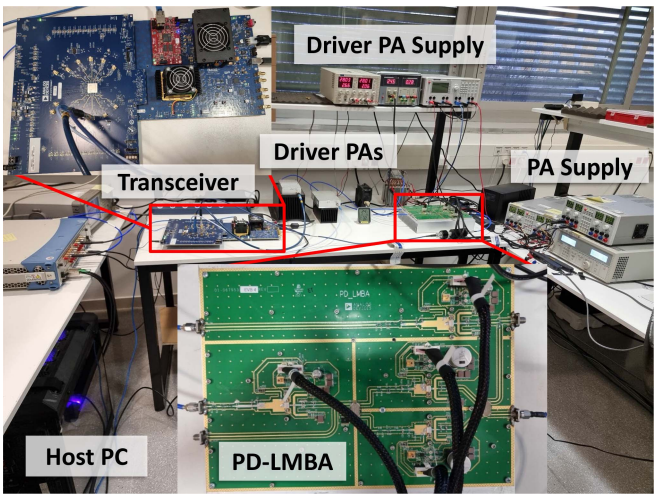


FIGURE 11. Picture of the PD-LMBA DUT test setup.

shown in Table 1, along with the target bias current through each transistor. The individual PAs were characterized using small signal tests and a pulsed CW signal with a 10% duty cycle. The PA performance was tuned by sweeping the matching capacitors values. The tuned PAs achieved a gain flatness of 2 dB over the design bandwidth.

The dual-input PD-LMBA architecture was chosen to overcome the power efficiency and bandwidth challenges for emerging multi-band cellular applications. In this design, it can operate over a broad frequency range and achieve over 50% efficiency at a large power backoff, satisfying the dual-band B3 (1805 MHz–1880 MHz) and B1 (2110 MHz–2170 MHz) use case requirements. However, the dual-input amplifier architecture is inherently more non-linear and complicated compared to the commonly used single-input Doherty amplifier architecture, necessitating advanced shaping functions and machine learning DPD algorithms to achieve the emission performance demanded by cellular infrastructure applications. In this work, the shaping function used to generate the control and main signals is described in [39], and it is designed to maximize power efficiency.

B. EXPERIMENTAL SET-UP

As shown in Fig. 11, experimental results were obtained using a Python-controlled test bench. The test bench features the Analog Devices ADRV903x RF agile transceiver evaluation platform and the Analog Devices dual-input PD-LMBA prototype. The ADRV903x is a highly integrated software-defined

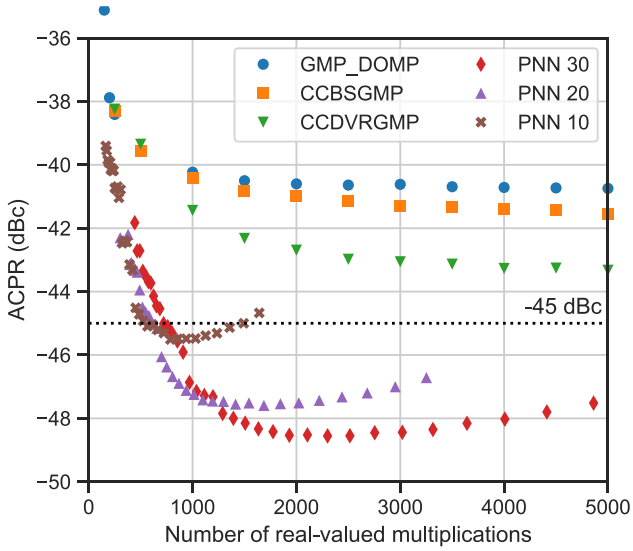


FIGURE 12. ACPR performance VS number of multiplications of the pruned PNN DPD model and other DPD models.

radio (SDR), optimized for the high performance and low power consumption demanded by cellular infrastructure applications. The SDR supports a wide range of RF frequencies between 600 MHz and 6000 MHz. The SDR transmitter delivers excellent in-band noise performance (i.e., −157 dBFS/Hz) while supporting a synthesis bandwidth of 800 MHz and a baseband input data rate up to 983.04 Msps.

The input stimulus consists of a 64-QAM pseudo-OFDM signal with an instantaneous bandwidth of 200 MHz operating at an RF frequency of 2 GHz for the experimental validation. To accommodate a DPD bandwidth expansion of 5x, the baseband sampling rate is chosen as 983.04 Msps. The PAPR of the input signal is limited to 8 dB by a crest factor reduction (CFR) technique from the family of clipping and filtering. With an average output power of 40 dBm, the power efficiency achieved is around 50%.

C. EXPERIMENTAL TEST RESULTS

Figs. 12 and 13 shows the linearization performance of the PNN-based DPD in terms of ACPR and NMSE, respectively, when applying pruning in the DLA identification loop. Because in ANNs the real-valued weights are dominating the number of parameters in comparison to the number of biases, we can assume that each parameter will require a real-valued multiplication to compute the DPD output. We considered two hidden layers for the evaluation of the PNN-based DPD. The first hidden layer is initialized with different number of neurons, i.e., $N_1 = 10$, 20 and 30, while for the second hidden layer, a fixed number of neurons is considered, i.e., $N_2 = 10$. The behavioral modeling and throughput performance have already been shown and discussed in Figs. 5 and 9, respectively. As observed in Fig. 12, without applying any pruning, the linearization performance improves with the number of hidden neurons, at the price of employing a large number of

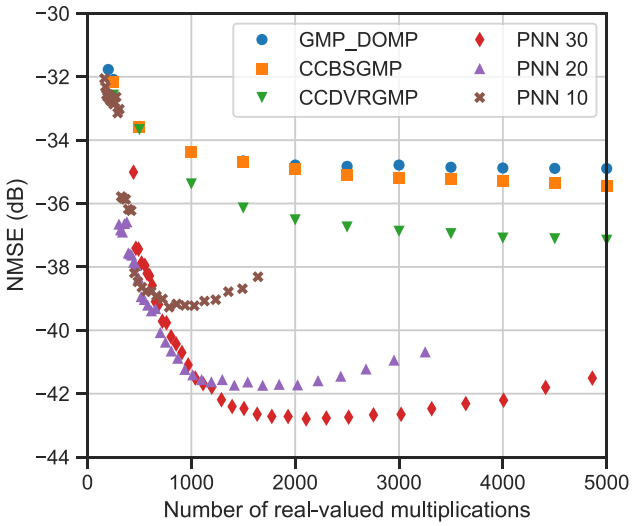


FIGURE 13. NMSE performance between the PA output and the generated signal of the pruned PNN DPD model and other DPD models.

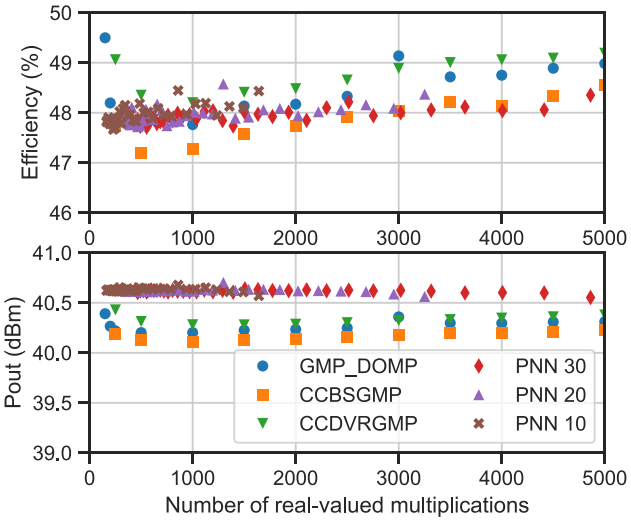


FIGURE 14. Efficiency and output power of the test cases.

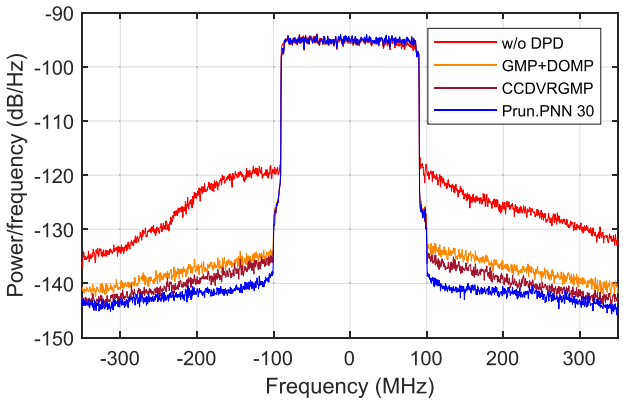


FIGURE 15. PA output spectra with and without linearization.

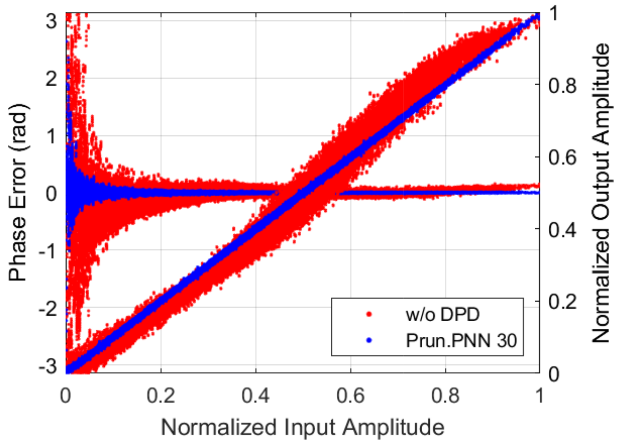


FIGURE 16. AM/AM and AM/PM of the PA output.

parameters (i.e, 1642 parameters for $N_1 = 10$; 3252 parameters for $N_1 = 20$ and 4862 parameters for $N_1 = 30$), which ultimately is translated into a large number of real-valued multiplications. By pruning the PNN model, the number of parameters can be reduced and the linearization performance can be further improved until the PNN DPD model starts losing its critical basis or features. When the number of multiplications is limited to 1000, the models initialized with a larger hidden layer size (i.e., PNN 20 and PNN 30) outperform the PNN 10 with a smaller initial size. Therefore, these extra hidden neurons are indeed meaningful for providing more linearization or modeling performance, and the pruning method is capable of preserving the most relevant features.

In order to justify the need for the ANN-based DPD models, we have also evaluated the linearization performance obtained with other popular DPD behavioral models. We have considered the GMP model, where its most relevant basis functions are selected using the DOMP algorithm, and two different cascaded (CC) models [40]. In particular, a two-stage CC B-Spline GMP (CCBSGMP) model and a two-stage CC DVR and GMP (CCDVRGMP) model. Note that these models are operated with complex-valued numbers. Typically, a basis function is computed by the product of a complex-valued parameter, a real-valued nonlinear feature, and a complex-valued delayed input. Thus, each parameter requires a minimum of 5 multiplication operations, while the PNN model only requires one multiplication for a weight parameter. Therefore, to fairly assess the DPD model complexity, in Figs. 12–14 the ACPR, NMSE, power efficiency and output power are evaluated for different number of real-valued multiplications as well as for different DPD models.

As depicted in Figs. 12 and 13, none of the GMP or CC models is able to meet the linearity requirements of the PD-LMBA when considering a 200 MHz OFDM-based signal. The GMP model with DOMP selected basis stops improving the linearization performance after 400 parameters (i.e., 2000 real-valued multiplications). The CC DPD models are capable of achieving better linearization performance with

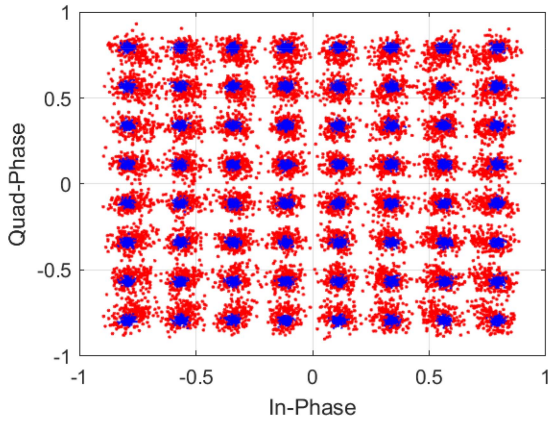


FIGURE 17. Constellation plot of the PA output with and without PNN linearization.

TABLE 2. PD-LMBA Linearization Performance

Model	NParam.	NMSE (dB)	ACPR (dBc)	Pout (dBm)	Eff. (%)	EVM (%)
w/o DPD	N/A	-19.8	-27.3	40.4	50.0	5.26
GMP+DOMP	1000	-34.9	-40.7	40.3	49.0	1.88
CCBSGMP	1000	-35.4	-41.6	40.2	48.6	1.89
CCDVRGMP	1000	-37.2	-43.3	40.4	49.2	1.82
Pruned PNN 10	939	-39.2	-45.5	40.6	48.0	1.77
Pruned PNN 20	938	-41.2	-47.1	40.6	47.9	1.75
Pruned PNN 30	909	-38.2	-45.1	40.6	50.1	1.67
		-38.4	-45.3	39.7	47.3	1.75

1000 parameters (i.e., 5000 real-valued multiplications) but still they cannot meet the requirement of ACPR<-45 dBc. As shown in [29], additional CC stages can be utilized to achieve ACPR values similar to those presented in this paper when using pruned PNNs. However, as demonstrated in [41], the computational complexity, measured in terms of real multiplications, is significantly lower when using the pruned PNNs proposed in this paper compared to the pruned N -stage CC models in [29]. Finally, Fig. 14 shows the power efficiency and output power for all the DPD models. The output power is kept quite stable around 40.5 dBm and the power efficiency is maintained around 48%. Therefore, independently on the DPD model used or the pruning strategy, only small variations of power efficiency ($< \pm 1$ percentage points) and mean output power (< 0.5 dB) are observed.

Table 2 shows the linearization performance in terms of NMSE, ACPR, EVM, output power and efficiency, comparing different DPD models with around 1000 parameters. As observed, all the pruned PNN models outperform the other Volterra-based and CC models and can meet the linearity requirement of ACPR < -45 dBc. To evaluate the robustness of the PNN DPD model, we adjusted the bias of the PD-LMBA slightly to squeeze a little bit more of power efficiency, what changed a bit the PA behaviour. Then, we retrained the PNN 30 model including data with 1dB of input power back-off to evaluate if the model was still capable of meeting the linearization specs. The last two rows in Table 2 show the linearization performance using the same model for the two different input power levels. As observed, the PNN 30 is robust enough to account for the input power back-off variations.

Fig. 15 shows the spectra of the 200 MHz modulated signal at the PA output without and with DPD linearization considering different DPD models. Finally, Figs. 16 and 17 show the AM/AM and AM/PM characteristics, and the 64-QAM constellation plot, respectively, of the PD-LMBA before and after DPD linearization considering the pruned PNN 30 model with 909 parameters.

V. CONCLUSION

In this paper, we have shown how ANN-based DPD is a feasible solution for real-time linearization of wideband high-efficiency PAs. On the one hand, we have proposed a pruning method based on ℓ_1 regularization that is applied to the first hidden layer of a PNN to reduce the number of parameters. On the other hand, specific details on the GPU implementation of the PNN-based DPD have been presented and discussed. Moreover, the throughput performance has been evaluated for different number of parameters of the PNN model.

The challenging linearization of the PD-LMBA DUT when operated with 200 MHz 5G-like test signals is only possible when considering PNN-based DPD. More conventional DPD models cannot fulfill the linearity requirement of ACPR $<$ -45 dBc. As for pruning strategy, we observed that better linearization performance is obtained when pruning from a PNN model with a larger initial number of parameters. Therefore, with around 900 parameters after pruning a PNN DPD model with an initial number of 4862 parameters it is possible to meet the linearity requirements with a mean output power around 40 dBm and 50% power efficiency. In addition, the GPU implementation of the PNN-based DPD is proved to be viable with a throughput around 1 GSa/Sec when considering around 1000 real-valued parameters.

ACKNOWLEDGMENT

The authors would like to thank A. Dubey, M. Cope, A. BenArfi and N. Outaleb from ADI for the donation and interesting discussion on the PD-LMBA used in this paper.

REFERENCES

[1] R. Pengelly, C. Fager, and M. Ozen, "Doherty's legacy: A history of the doherty power amplifier from 1936 to the present day," *IEEE Microw. Mag.*, vol. 17, no. 2, pp. 41–58, Feb. 2016.

[2] C. Kantana, M. Benosman, R. Ma, and Y. Komatsuzaki, "A system approach for efficiency enhancement and linearization technique of dual-input doherty power amplifier," *IEEE J. Microwaves*, vol. 3, no. 1, pp. 115–133, Jan. 2023.

[3] A. Pitt, G. Jindal, K. Morris, and T. Cappello, "A broadband asymmetrical Doherty power amplifier with optimized continuous mode harmonic impedances," *IEEE J. Microwaves*, vol. 3, no. 4, pp. 1120–1133, Oct. 2023.

[4] R. Quaglia, J. Pang, S. C. Cripps, and A. Zhu, "Load-modulated balanced amplifier: From first invention to recent development," *IEEE Microw. Mag.*, vol. 23, no. 12, pp. 60–70, Dec. 2022.

[5] Z. Popovic, "Amping up the PA for 5G: Efficient GaN power amplifiers with dynamic supplies," *IEEE Microw. Mag.*, vol. 18, no. 3, pp. 137–149, May 2017.

[6] W. Li, G. Montoro, and P. L. Gilabert, "Digital linearization of wideband envelope tracking power amplifiers for mobile terminals," *IEEE Trans. Microw. Theory Tech.*, vol. 71, no. 1, pp. 48–58, Jan. 2023.

[7] T. Wang, W. Li, R. Quaglia, and P. L. Gilabert, "Machine-learning assisted optimisation of free-parameters of a dual-input power amplifier for wideband applications," *Sensors*, vol. 21, no. 8, 2021, Art. no. 2831. [Online]. Available: https://www.mdpi.com/1424-8220/21/8/2831

[8] P. Roblin, C. Quindroit, N. Naraharisetti, S. Gheitanchi, and M. Fitton, "Concurrent linearization: The state of the art for modeling and linearization of multiband power amplifiers," *IEEE Microw. Mag.*, vol. 14, no. 7, pp. 75–91, Nov.–Dec. 2013.

[9] C. Fager, T. Eriksson, F. Barradas, K. Hausmair, T. Cunha, and J. C. Pedro, "Linearity and efficiency in 5G transmitters: New techniques for analyzing efficiency, linearity, and linearization in a 5G active antenna transmitter context," *IEEE Microw. Mag.*, vol. 20, no. 5, pp. 35–49, May 2019.

[10] D. López-Bueno, G. Montoro, and P. L. Gilabert, "Training data selection and dimensionality reduction for polynomial and artificial neural network MIMO adaptive digital predistortion," *IEEE Trans. Microw. Theory Techn.*, vol. 70, no. 11, pp. 4940–4954, Nov. 2022.

[11] D. Scheurs, M. O'Droma, A. A. Goacher, and M. Gadringer, Eds., *RF Power Amplifier Behavioural Modeling*. Cambridge, U.K.: Cambridge Univ. Press, 2009.

[12] J. Kim and K. Konstantinou, "Digital predistortion of wideband signals based on power amplifier model with memory," *Electron. Lett.*, vol. 37, no. 23, pp. 1417–1418, Nov. 2001.

[13] D. R. Morgan, Z. Ma, J. Kim, M. G. Zierdt, and J. Pastalan, "A generalized memory polynomial model for digital predistortion of RF power amplifiers," *IEEE Trans. Signal Process.*, vol. 54, no. 10, pp. 3852–3860, Oct. 2006.

[14] A. Zhu, "Decomposed vector rotation-based behavioral modeling for digital predistortion of RF power amplifiers," *IEEE Trans. Microw. Theory Techn.*, vol. 63, no. 2, pp. 737–744, Feb. 2015.

[15] A. Molina, K. Rajamani, and K. Azadet, "Digital predistortion using lookup tables with linear interpolation and extrapolation: Direct least squares coefficient adaptation," *IEEE Trans. Microw. Theory Techn.*, vol. 65, no. 3, pp. 980–987, Mar. 2017.

[16] P. L. Gilabert, A. Cesari, G. Montoro, E. Bertran, and J.-M. Dilhac, "Multi-lookup table FPGA implementation of an adaptive digital predistorter for linearizing RF power amplifiers with memory effects," *IEEE Trans. Microw. Theory Techn.*, vol. 56, no. 2, pp. 372–384, Feb. 2008.

[17] T. Liu, S. Boumaiza, and F. Ghannouchi, "Dynamic behavioral modeling of 3G power amplifiers using real-valued time-delay neural networks," *IEEE Trans. Microw. Theory Techn.*, vol. 52, no. 3, pp. 1025–1033, Mar. 2004.

[18] M. Rawat, K. Rawat, and F. M. Ghannouchi, "Adaptive digital predistortion of wireless power amplifiers/transmitters using dynamic real-valued focused time-delay line neural networks," *IEEE Trans. Microw. Theory Techn.*, vol. 58, no. 1, pp. 95–104, Jan. 2010.

[19] M. Rawat and F. M. Ghannouchi, "A mutual distortion and impairment compensator for wideband direct-conversion transmitters using neural networks," *IEEE Trans. Broadcast.*, vol. 58, no. 2, pp. 168–177, Jun. 2012.

[20] T. Liu et al., "Rf power amplifier modeling and linearization with augmented RBF neural networks," in *Proc. IEEE Int. Workshop Electromagn., Appl. Student Innov. Competition*, 2016, pp. 1–3.

[21] F. Mkadem and S. Boumaiza, "Physically inspired neural network model for RF power amplifier behavioral modeling and digital predistortion," *IEEE Trans. Microw. Theory Techn.*, vol. 59, no. 4, pp. 913–923, Apr. 2011.

[22] E. Guillena, W. Li, G. Montoro, R. Quaglia, and P. L. Gilabert, "Reconfigurable DPD based on ANNs for wideband load modulated balanced amplifiers under dynamic operation from 1.8 to 2.4 GHz," *IEEE Trans. Microw. Theory Techn.*, vol. 70, no. 1, pp. 453–465, Jan. 2022.

[23] D. Wang, M. Aziz, M. Helaoui, and F. M. Ghannouchi, "Augmented real-valued time-delay neural network for compensation of distortions and impairments in wireless transmitters," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 1, pp. 242–254, Jan. 2019.

[24] X. Hu et al., "Convolutional neural network for behavioral modeling and predistortion of wideband power amplifiers," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 8, pp. 3923–3937, Aug. 2022.

[25] A. Fischer-Bühner, A. Brihuega, L. Anttila, M. D. Gomony, and M. Valkama, "Mixture of experts neural network for modeling of power amplifiers," in *Proc. IEEE/MTT-S Int. Microw. Symp.*, 2022, pp. 510–513.

[26] A. Fischer-Bühner, L. Anttila, M. D. Gomony, and M. Valkama, "Phase-normalized neural network for linearization of RF power amplifiers," *IEEE Microw. Wireless Technol. Lett.*, vol. 33, no. 9, pp. 1357–1360, Sep. 2023.

[27] A. Barry, W. Li, J. A. Becerra, and P. L. Gilabert, "Comparison of feature selection techniques for power amplifier behavioral modeling and digital predistortion linearization," *Sensors*, vol. 21, no. 17, 2021, Art. no. 5772. [Online]. Available: https://www.mdpi.com/1424-8220/21/17/5772

[28] J. A. Becerra, M. J. Madero-Ayora, J. Reina-Tosina, C. Crespo-Cadenas, J. Garcia-Frias, and G. Arce, "A doubly orthogonal matching pursuit algorithm for sparse predistortion of power amplifiers," *IEEE Microw. Wireless Compon. Lett.*, vol. 28, no. 8, pp. 726–728, Aug. 2018.

[29] R. Criado, W. Li, W. Thompson, G. Montoro, K. Chuang, and P. L. Gilabert, "Model-order reduction of multistage cascaded models for digital predistortion," *IEEE J. Microwaves*, vol. 5, no. 1, pp. 137–149, Jan. 2025.

[30] H. Barkhordar-Pour, J. G. Lim, M. Almoneer, P. Mitran, and S. Boumaiza, "Real-time FPGA-based implementation of digital predistorters for fully digital MIMO transmitters," in *IEEE MTT-S Int. Microw. Symp. Dig.*, 2023, pp. 263–266.

[31] W. Li, N. Bartzoudis, J. Rubio Fernández, D. López-Bueno, G. Montoro, and P. L. Gilabert, "FPGA implementation of a linearization system for wideband envelope tracking power amplifiers," *IEEE Trans. Microw. Theory Techn.*, vol. 71, no. 4, pp. 1781–1792, Apr. 2023.

[32] S. Wang, M. Roger, J. Sarrazin, and C. Lelandais-Perrault, "Augmented iterative learning control for neural-network-based joint crest factor reduction and digital predistortion of power amplifiers," *IEEE Trans. Microw. Theory Techn.*, vol. 68, no. 11, pp. 4835–4845, Nov. 2020.

[33] D. López-Bueno, P. L. Gilabert, and G. Montoro, "Dataset reduction for neural network based digital predistorters under strong nonlinearities," in *Proc. IEEE Topical Conf. RF/Microw. Power Amplifiers Radio Wireless Appl.*, 2021, pp. 8–11.

[34] J. Chani-Cahuana, P. N. Landin, C. Fager, and T. Eriksson, "Iterative learning control for RF power amplifier linearization," *IEEE Trans. Microw. Theory Techn.*, vol. 64, no. 9, pp. 2778–2789, Sep. 2016.

[35] NVIDIA Corporation, "CUDA C programming guide," 2023. [Online]. Available: https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html

[36] W. Li, G. Montoro, and P. L. Gilabert, "GPU versus FPGA implementation of a digital predistortion linearizer for wideband radiofrequency power amplifiers," *AEU Int. J. Electron. Commun.*, vol. 174, 2024, Art. no. 155040. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1434841123005149

[37] J. Pang et al., "Analysis and design of highly efficient wideband rfRFinput sequential load modulated balanced power amplifier," *IEEE Trans. Microw. Theory Techn.*, vol. 68, no. 5, pp. 1741–1753, May 2020.

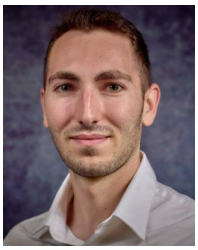
[38] Y. Cao and K. Chen, "Pseudo-Doherty load-modulated balanced amplifier with wide bandwidth and extended power back-off range," *IEEE Trans. Microw. Theory Techn.*, vol. 68, no. 7, pp. 3172–3183, Jul. 2020.

[39] W. Li, G. Montoro, W. Thompson, K. Chuang, and P. L. Gilabert, "Digital shaping and linearization of a dual-input load-modulated balanced amplifier," in *Proc. Int. Workshop Integr. Nonlinear Microw. Millimetre-Wave Circuits*, 2023, pp. 1–3.

[40] S. Wang, M. A. Hussein, O. Venard, and G. Baudoin, "Optimal sizing of cascaded digital predistortion for linearization of high power amplifiers," in *Proc. IEEE Asia Pacific Microw. Conf.*, 2017, pp. 829–832.

[41] R. Criado, W. Li, W. Thompson, G. Montoro, K. Chuang, and P. L. Gilabert, "ANN versus cascaded behavioral models for DPD linearization of wideband dual-input PAs," in *Proc. IEEE Topical Conf. RF/Microw. Power Amplifiers Radio Wireless Appl.*, 2025, pp. 1–4.

WANTAO LI was born in Canton, China, in 1995. He received the B.E. degree in electronic information science and technology from Beijing Union University, Beijing, China, in 2017, and the M.S. degree in unmanned aircraft systems in 2020 from the Universitat Politècnica de Catalunya (UPC), Barcelona, Spain, where he currently working toward the Ph.D. degree with the Department of Signal Theory and Communications. In 2020, he joined the Components and Systems for Communications (CSC) Research Group. His research interests include signal processing for communication systems, power amplifier linearization for MISO PAs, and the efficient RTL implementations in FPGA.



RAÚL CRIADO (Student Member, IEEE) received the double B.E. degree in telecommunications systems engineering and aerospace systems engineering from the Universitat Politècnica de Catalunya, Barcelona, Spain, in 2024, where he is currently working toward the M.Sc. degree in aerospace science and technology. Since 2023, he has been involved in a research program with Universitat Politècnica de Catalunya. His research focuses on digital signal processing techniques for the control and linearization of high-efficiency RF power amplifiers.

WILLIAM THOMPSON received the M.Eng. degree in electronic engineering from the University of Southampton, Southampton, U.K., and the Ph.D. degree with the University of Bristol, Bristol, U.K. He is currently a Principal Engineer with Analog Devices, Inc., Bath, U.K., specializing in signal processing of telecommunications signals, working on a wide range of projects including 5G ORAN and efficient PA systems. He has worked as a Postdoctoral Researcher with the University of Bristol and then as a research engineer with Toshiba, before moving to the Bath office of ADI.

GABRIEL MONTORO received the M.Sc. degree in telecommunication engineering and the Ph.D. degree from the Universitat Politècnica de Catalunya (UPC), Barcelona, Spain, in 1990 and 1996, respectively. In 1991, he joined the Department of Signal Theory and Communications (TSC), where he is currently an Associate Professor. His first research works were done on the area of adaptive control. His research interests mainly focuses on the use of signal processing strategies for power efficiency improvement in communications systems.

KEVIN CHUANG (Senior Member, IEEE) received the B.S. degree from the University of California, Santa Barbara (UCSB), Santa Barbara, CA, USA, and the M.S. and Ph.D. degrees from Georgia Institute of Technology, Atlanta, GA, USA. He is currently a Technology Leader with Analog Devices, Inc., Wilmington, MA, USA, where he spearheads radio technology innovations and development for 6G wireless communications with the intersection of RF systems and signal processing. He also leads the wireless standards activities for ATIS/Next G Alliance and 3GPP. Prior to joining ADI, he co-founded NanoSemi, a spin-off from the MIT and MIT Lincoln Laboratory, where he contributed to the development of wideband linearization technologies for various wireless systems from concept to product launch, which was subsequently acquired by MaxLinear. He held wireless systems and RF/mixed-signal IC design positions with Wafer LLC, MIT Lincoln Laboratory and Samsung Electro-Mechanics in Beverly, MA, Lexington, MA, and Atlanta, GA, respectively.

PERE L. GILABERT (Senior Member, IEEE) received the M.Sc. degree in telecommunication engineering and the and the Ph.D. degree from the Universitat Politècnica de Catalunya (UPC), Barcelona, Spain, in 2002 and 2008, respectively. He developed his master's thesis with the University of Rome "La Sapienza," Rome, Italy, with an Erasmus Exchange Grant. In 2003, he joined the Department of Signal Theory and Communications. He is currently an Associate Professor with the Castelldefels School of Telecommunications and Aerospace Engineering. His research interests include the field of linearization techniques and digital signal processing solutions for highly efficient transmitter architectures. He was the recipient of the Extraordinary Doctoral Prize for his Ph.D. degree.