Selective Method with Adapted Activation Function and Applied in Non-linear Optimization for Modeling of Power Amplifiers

ANA PAULA PRINCIVAL MACHADO
dept. Electrical Engineering (UFPR)
Universidade Federal do Paraná
Curitiba, Brazil
anaprincival@ufpr.br

EDUARDO GONÇALVES DE LIMA
dept. Electrical Engineering (UFPR)
Universidade Federal do Paraná
Curitiba, Brazil
elima@eletrica.ufpr.br

***Abstract*—Neural networks are an increasingly used data manipulation method, becoming popular due to their versatility and efficiency. The present work is based on the Group Method of Data Handling (GMDH) model of neural network to manipulate the data coming from the power amplifier (PA). The PA is a device used in electrical and electronic circuits, with the purpose of increasing the signal applied to its input. This needs to operate linearly to optimize the system energy, and when its signal reaches higher orders, linearization methods are used to correct it. Previously, two codes were developed with the objective of linearizing the PA signal, and were submitted to different scenarios for analysis of their performance. The most efficient of them is called Selective, as it is more careful in the selection of neurons that will remain active in the network. In the present work, the Selective is submitted to the non-linear optimization method with alterations in the activation function in relation to its original conception. The Selective is transformed into a real network, with the division of the activation function according to the neuron inputs, and submitted to MATLAB software optimization. This, in turn, presents Normalized Mean Square Error (NMSE) in the order of -29 dB, which is a small error value, although greater than the original conditions. The analysis is consolidated as a good performance, although not great, with potential for improvements in the manipulation of PA data.**

Keywords—Power amplifiers, real neural network, Group Method of Data Handling, non-linear optimization.

I. INTRODUCTION

Neural networks are being increasingly used due to their dynamism and efficiency. Currently they are widely used in computer vision, natural language processing, machine learning, games, simulations and robotics. In the present work, they are used in data manipulation in order to linearize the power amplifier (PA) signal.

PAs are devices commonly used in the telecommunications area and aim to increase the signal applied to its input, be it voltage or current. They consist of transmitter, medium and receiver, and must operate in linear order to optimize the energy of the system in which they operate. In cases where the signal reaches orders greater than one (wireless communication networks and TV broadcast systems, for example), more power is required from the system and linearization methods are used to correct it [1].

Among the existing neural networks, those of the Feed Forward type stand out, which take the signal from the input to the output in a unidirectional way. Still in this topology, the Group Method of Data Handling (GMDH) is approached, which is a pioneering method in the autonomy of the neural network. The GMDH stood out for its ability to automatically select variables, self-organizing structure, reduction of overfitting, in addition to easy interpretability [2]. The model can be applied in time series forecasting (finance, economy, energy, climate), demand analysis, and systems diagnosis, for example.

Thus, two GMDH-based codes were developed, which are essentially neural networks built to linearize the PA signal, differing in their selection method regarding the neurons that will continue in the network [3]. They consist of three layers, each with a certain number of neurons, each having an activation function, receiving two inputs and resulting in one output. At the end of the network, the Normalized Mean Square Error (NMSE) is calculated, which measures the efficiency of the structure. The code titled Selective is more rigorous, starting the selection from the first layer, while the Embracing code only does it from the second layer. In this way, the Selective presents more efficient results and has more advantages to be explored in other scenarios.

With this, the work begins with the presentation of the Selective method, in Section II. Then, in Section III it is shown the adaptation of the Selective to a real neural network, which was previously built and certified as to its efficiency [4]. In Section IV, the application of real Selective in the non-linear optimization method is discussed, which is a tool that considers the dynamics of the network in real time and leads to the lowest NMSE [5]. Finally, the results obtained and the conclusion are presented.

II. SELECTIVE METHOD

The original Selective method is composed of three layers, where the output of each layer is used as input to the next layer. The network is self-organizing, so the initial number of *In* inputs chosen will define the number of neurons in the first layer, *F*, as

$$F = \frac{In!}{2!(In - 2)!}. \quad (1)$$

The second layer has two neurons, while the third has a single neuron, which produces the final output obtained from the network used in the calculation of the NMSE (dB), given by

$$NMSE = 10\log \frac{\sum_{n=1}^{N} |e(n)|^2}{\sum_{n=1}^{N} |yref(n)|^2}, \quad (2)$$

where $e(n)$ is the difference between the desired and obtained outputs, $y(ref)$ is the desired output, n is each time instant and N is the total number of samples.

Each neuron has an activation function, responsible for providing the necessary coefficients to assemble the outputs. As the neuron receives two inputs (x_1 and x_2) and results in an output $y(x)$, the function must involve both inputs. Originally, the extraction of coefficients is done using Least Squares (LS), but the method can be changed according to the scenario to which the Selective is submitted.

In this work, the activation function $y(x)$ is separated into two: $g(x_1)$ and $g(x_2)$, one for each respective input of the neuron, as

$$g(x_1) = a_0 + a_1x_1 + a_1x_1^2 \quad (3)$$

and

$$g(x_2) = b_0 + b_1x_2 + b_1x_2^2. \quad (4)$$

At the end of processing, (3) and (4) are multiplied together, resulting in the total output of the neuron. The illustration of the topology of the Selective can be seen in Fig. 1.

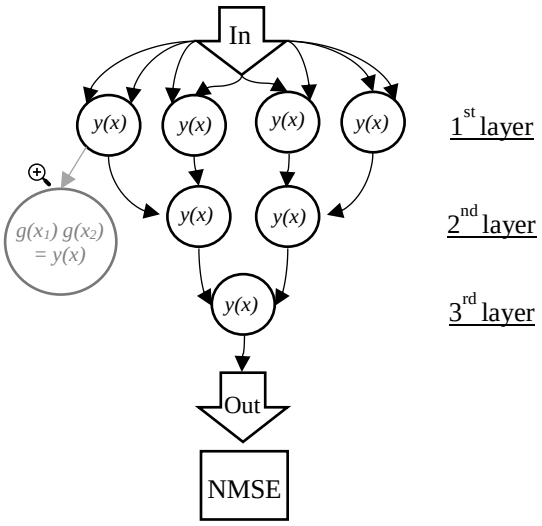

Fig. 1 – Selective Method Structure.

As the first layer has a significantly larger number of neurons than the second, it is necessary to develop a method that tests all organization possibilities. This is done by calculating the combination of 2 by 2 of all neurons contained in F, so that the whole set is tested as network continuity. Thus, the Selective is executed the necessary number of times to test all possible combinations. At the end of each routine, the NMSE is calculated and stored, so that at the end of the tests the smallest error can be identified.

III. Selective Method Adapted for Real Neural Network

Originally, the constructed network uses a complex set of samples, derived from the PA data. However, the Selective was subjected to the real neural network scenario [4], so that the initial complex set was transformed into a real set.

This real set is based on matrices with the number of columns corresponding to the number of past instants being referenced, using the magnitude and angle of the respective complex samples. For example, for $m = 1$, the columns are a_n, a_{n-1}, $cos(\theta_n - \theta_{n-1})$ and $sin(\theta_n - \theta_{n-1})$, where n is the instant of the complex vector, a is the amplitude, and θ is the angle of the complex sample. For $m = 2$, the columns are a_n, a_{n-1}, a_{n-2}, $cos(\theta_n - \theta_{n-1})$, $sin(\theta_n - \theta_{n-1})$, $cos(\theta_{n-1} - \theta_{n-2})$ and $sin(\theta_{n-1} - \theta_{n-2})$. The higher the value of m, the more past instants are referenced, and more columns are added to the matrix. In the work carried out [4] m is varied from 1 to 4, which makes the number of columns h of the actual entry vary from 4 to 13. A simple organization scheme is shown in Fig. 2.

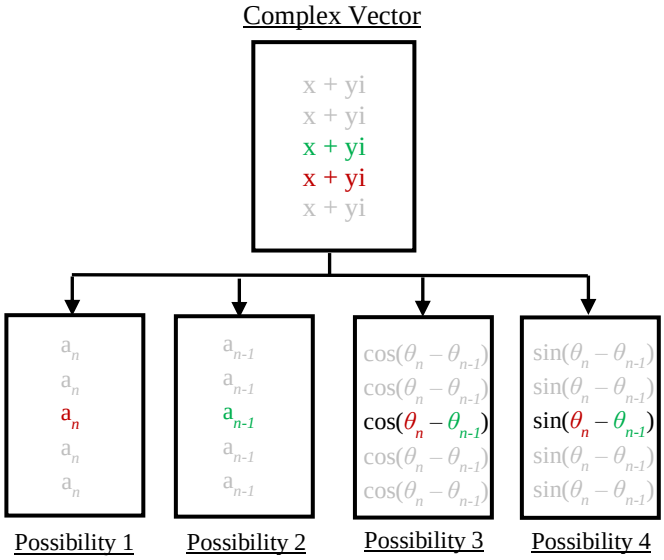

Fig. 2 – Organization scheme of real matrices.

Then, the value of In is chosen to define the number of initial inputs of the neural network, which commonly varies from 1 to 4. Each value of In has all the values of m available, so that there are many possibilities of combination. At each fixed value of m, combinations of In and In of the value of h are tested in order to test all possibilities and verify which has the best performance.

The real architecture relies on two networks, which are essentially the Selective method, but which differ in the desired output. The first network is called In-Phase (I), and has the target as $b_n cos(\alpha_n - \theta_n)$, responsible for the real part. The second network is called Quadrature (Q), whose target is $b_n sin(\alpha_n - \theta n)$, responsible for the imaginary part. In the targets, n is the referenced instant, b is the modulus of the output sample, α is

the angle of the output sample and θ is the angle of the input sample. At the end of both processing, the responses are recombined to become a complex vector again, to be used in the calculation of the NMSE. Fig. 3 shows the organization of this network.

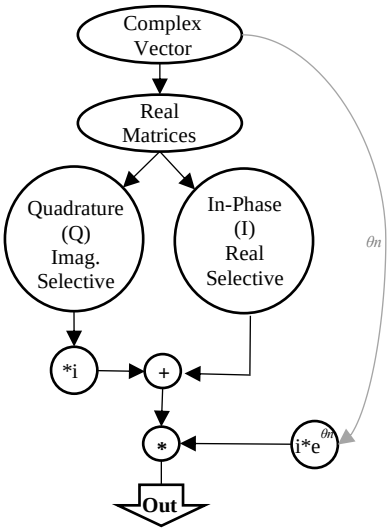

Fig. 3 – Organization of the Real Selective Method.

IV. Application of Non-linear Optimization

There are different methods of extracting the coefficients in the scenarios to which the Selective is submitted, and one of them is the non-linear optimization.

It is a system that considers the network dynamism simultaneously, so that it extracts the coefficients all at once and is able to induce better results [5]. It looks for a minimum error, but which is never zeroed out, due to the fact that there are more equations than unknowns.

In MATLAB, the non-linear optimization method is set by the *lsqnonlin* command. For the execution it is necessary a function and an initial parameter, making multiple iterations until finding the minimum error. In this case, the function used was the best scenario (which resulted in the best NMSE) of the Selective adapted to the real network, with the activation function divided for each respective input. The stopping criterion used was the difference, in modulus, of the output obtained and the desired output of the network. As initial parameter, a random vector was used to obtain new coefficients of all layers, extracted with non-linear optimization. Fig. 4 exemplifies how the method works.

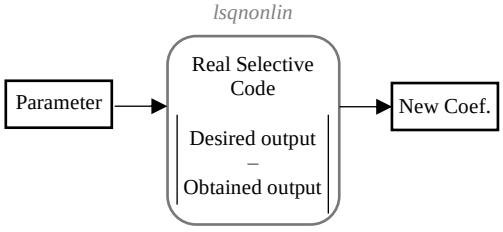

Fig. 4 – Selective Method Structure.

V. Simulation Results

All mentioned structures are built and tested in MATLAB software, and the sample set used in the development is a GaN technology power amplifier, with HEMT semiconductor, carrier frequency of 900 MHz and envelope signal of 3.84 MHz of bandwidth. The extraction set has 3221 positions, while the validation set has 2001 positions [6].

The best scenario of Selective adapted to the real network is configured by applying $In = 4$ and $m = 3$ [4]. When identifying the best scenario, to simplify the simulations, four neurons were fixed in the first layer, two in the second and a single one in the third, totaling seven neurons and 42 coefficients in the network. The input vectors of the original codes were directly inserted into the simulated network, focusing on extracting the coefficients from this best scenario already identified. As the *lsqnonlin* command does not allow the initial parameter to be of type *complex*, it is necessary to pass 84 values of type *double*, which will be divided and transformed into 42 complex values within the function. For this, MATLAB's *rand* command is used, which generates a random vector of a certain size whenever the code is executed.

The function is essentially the Selective best-case scenario and is what *lsqnonlin* runs on top of. About 1000 iterations are stipulated for each execution, and at the end of all of them the new coefficients found are returned. Then these are inserted directly into the body of the code, in order to extract the NMSE for extraction and validation. Ten routines of 1000 iterations each were trained, storing the respective error value. The results obtained are shown in Table I.

	Iterations	NMSE Extraction (dB)	NMSE Validation (dB)
1	1000	-29.2575	-29.7756
2	1000	-30.1601	-30.8195
3	1000	-29.0820	-29.7918
4	1000	-29.2575	-29.7756
5	1000	-29.2575	-29.7756
6	1000	-30.1601	-30.8195
7	1000	-29.2575	-29.7756
8	1000	-30.1601	-30.8195
9	1000	-29.2575	-29.7756
10	1000	-29.2575	-29.7756

Table I – Results of iterations.

It is noticeable that the extraction NMSE are around -29 dB, and the validation NMSE are close to this value as well. This shows that during training there was no overfitting problem, so the model is well-fitted for both extraction and validation sets.

Furthermore, it is also noted that despite the different routines, some values are repeated in the Table I, even with the *rand* command, which generates random initial values. This can be explained by the fact that *lsqnonlin*, during training, returns to the same local minima, which can be global or not.

Although the error values are low, they are not as good as the initial tests with the real network [4] and with the non-linear

optimization [5], which presented NMSE around -35 dB. This may have occurred due to changes implemented in the activation function, which is now separated in two. The fact that the function is separated for each input of the neuron means that the extracted coefficients (directly correlated with the activation function) are not as optimized for the network as before, so that the best option turns out to be the one that considers both entries in the same activation function.

Although not in its best scenario, the Selective still performs well in general, presenting the obtained output close to the desired output. The Fig. 5 shows about forty samples of the amplitudes of desired output (extraction set) and obtained one. The scenario to which it was submitted is favorable for improvements, proving to be a good option for handling PA data.

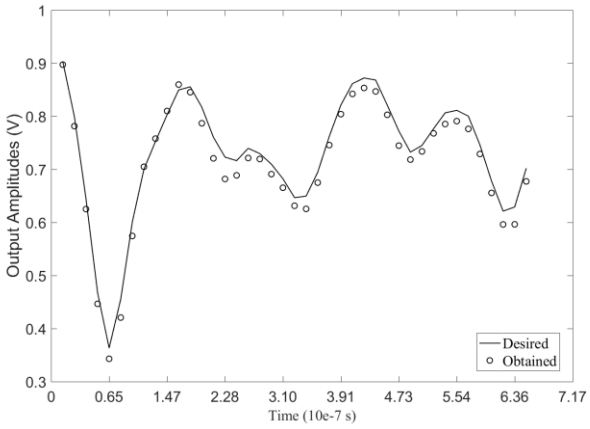

Fig. 5 – Output Amplitudes.

VI. Conclusions

Power amplifiers are devices used in electrical media in order to amplify the received signal. They need to operate linearly to maintain the energy efficiency of the system in which they operate. When this does not occur, a correction must be implemented.

The present work presents data manipulation through neural networks, more specifically those based on GMDH, which is a method that stands out for its advantages of accuracy and precision. During the research carried out, two codes based on GMDH were developed with the objective of linearizing the PA signal. These were subjected to different scenarios, so that their performance was analyzed. In general, the more rigorous structure in the selection of neurons, called Selective, presents better results, having greater chances of implying improvements in the continuation of research.

With that, the Selective method was adapted to a real neural network, whose activation function contained in each neuron was divided for each of its respective inputs (differently from the original model). The structure was then applied in the non-linear optimization method, possible through a simple command in the MATLAB software, which is capable of extracting all the network coefficients simultaneously.

Several tests were performed, 1000 iterations each, in order to analyze the obtained NMSE. The results show that the extraction and validation errors are concentrated around –29 dB, which is considered a small value, although greater than that obtained with the original model. Even without overfitting and ill-conditioning problems, the model with a split activation function does not perform as well as the one that considers both inputs at the same time. However, it still consolidates itself as a potential model for improvements in the handling of PA data.

To try to improve performance, it is possible to invest in increasing the extracted coefficients, directly modifying the activation function to include a cubed term. The advantage of this new activation function is its lower computational cost implementation through the use of 2 Look-Up Tables.

Acknowledgment

The authors would like to acknowledge the financial support provided by National Council for Scientific and Technological Development (CNPq) under the Program PIBIC UFPR 2022.

References

[1] ZANELLA, A. F.; LIMA, E. G. A GMDH based approach for the Behavioral Modeling of Radiofrequency Power Amplifiers. In: 19o Simpósio Brasileiro de Micro-ondas e Optoeletrônica e 14o Congresso Brasileiro de Eletromagnetismo, 2020. p. 266-270.

[2] IVAKHNENKO, A. G. The group method of data handling in long-range forecasting. Technological Forecasting and Social Change,v. 12, 2-3 1978.

[3] MACHADO, A. P. P.; LIMA, E. G. . Selective Algorithm for Group Method of Data Handling Applied to Power Amplifier Modeling. In: XXI Microelectronics Students Forum, 2021, Campinas. Proceedings of the XXI Microelectronics Students Forum, 2021. p. 1-4

[4] MACHADO, A. P. P.; LIMA, E. G. . Real-valued Neural Network Based in Group Method of Data Handling Applied to Power Amplifier Modeling. In: XXII Microelectronics Students Forum, 2022, Minuano. Proceedings of the XXII Microelectronics Students Forum, 2022. p. 1-4.

[5] MACHADO, A. P. P.; LIMA, E. G. . Algoritmo Seletivo com Diferentes Métodos de Extração para Modelagem de Amplificador de Potência Baseada em GMDH. In: Seminários de Microeletrônica do Paraná, 2022, Curitiba. Anais da Semicro, 2022. p. 1-4.

[6] BONFIM, E.; LIMA, E. G. A modified two dimensional Volterra-based series for the low-pass equivalent behavioral modeling of RF power amplifiers. Progress In Electromagnetics Research M, Vol. 47, 27-35, 2016.