Analog System Modeling Based on a Double Modified Complex Valued Neural Network

Antonio Luchetta, Stefano Manetti, and Maria Cristina Piccirilli

***Abstract*—The aim of this work is to present a novel technique for the identification of lumped circuit models of general distributed apparatus and devices. It is based on the use of a double modified complex value neural network. The method is not oriented to a unique class of electromagnetic systems, but it gives a procedure for the complete validation of the approximated lumped model and the extraction of the electrical parameter values. The inputs of the system are the geometrical (and/or manufacturing) parameters of the considered structure, while the outputs are the lumped circuit parameters. The method follows the Frequency Response Analysis (FRA) approach for elaborating the data presented to the network.**

I. Introduction

During the analog circuit design process, in many cases the phase of modelization, parameter identification and simulation of distributed circuits is still an arduous challenge. The difficulties are inherent in locating the parameters that can be extracted and in obtaining the requested precision for them. On the other hand, this phase can become essential to solve several problems, as, for example, the study of the transient part of the response, the evaluation of the electromagnetic compatibility, the estimate of the harmonic content, the detection and location of faults, the influence of a single parameter over the output final behavior. In the last few years several soft computing algorithms dealing with this subject have been developed, using artificial neural networks (ANNs) [1-3], genetic algorithms (GAs) [4,5], particle swarm optimizers (PSOs) [6,7].

In this paper we propose a new neural architecture able to accomplish the identification task. It is constituted by the union of a pair of multi-valued neuron neural networks with complex weights [8]. In this kind of architecture it is possible to use a set of multifrequency measurements or simulations made on the device, taken at different values of geometrical parameters, and train a MultiLayer Multi-Valued Neuron Network (MLMVN), able to estimate the electrical parameters of the lumped model. A single network is not sufficient to achieve the goal, because, in general, the supervised reference of the lumped model is not available. In order to overcome this limitation, another network must be added, having the role of approximating and "inverting" a network function of the chosen lumped model. The output of this layer consists of the numerical estimate of the electrical parameters of the device under analysis. The two neural networks are connected in cascade, in order to extract the best possible estimation of the electrical parameters of the equivalent lumped model. After the "two-phase" learning phase, the network is able to dimension the equivalent electrical model, for each different set of geometric parameters presented to the input. A further aspect to take into account for the correct identification of the model is the "solvability" of the model with respect to the parameters chosen for the identification. To this aim, a preliminary evaluation of the testability of the lumped model is performed for determining the solvability degree with respect to the circuit parameters [9]. If the proposed lumped model is not suitable or not complete, the neural learning process does not converge. Hence this kind of approach represents a useful information for the design process in the modeling and simulation phase.

A. Luchetta is with the Dipartimento di Ingegneria dell'Informazione (DINFO) of the University of Florence, Via S.Marta, 3, 50139 Firenze, Italy (phone: +39-055-4796461; fax: +39-055-4796442; e-mail: luchetta@ unifi.it).

S. Manetti is with the Dipartimento di Ingegneria dell'Informazione (DINFO) of the University of Florence, Via S.Marta, 3, 50139 Firenze, Italy (e-mail: stefano.manetti@ unifi.it).

M.C. Piccirilli is with the Dipartimento di Ingegneria dell'Informazione (DINFO) of the University of Florence, Via S.Marta, 3, 50139 Firenze, Italy (e-mail: mariacristina.piccirilli@ unifi.it).

II. Theoretical Foundation

In this section a synthetic theoretical description of the modified neural network is presented.

A. The multi-valued neuron (MVN)

The discrete multi-valued neuron (MVN) was presented in [8] as a neural element based on the principles of multiple-valued threshold logic in the field of complex numbers. These principles were formulated in [10], then developed and generalized in [8], and recently summarized and extended to the last results in [11]. The continuous version of MVN used in this work performs a mapping between n inputs and a single output. This mapping is described by a multi-valued (k-valued) function of n variables $f(x_1,\ldots,x_n)$, that, in the limit-case of continuous type of activation function, has its inputs and output located on the unit circle. For the continuous MVN the activation function is:

$$P(z) = e^{i\mathrm{Arg}\, z} = z/|z| \tag{1}$$

where $z=w_0+w_1x_1+\ldots\ w_nx_n$ is the weighted sum, and $\mathrm{Arg}(z)$ is the main value of the argument (phase) of the complex number z. Thus, for the continuous MVN, the output is the projection of the weighted sum on the unit circle, as it is

determined by the activation function (1) (see Fig. 1). The MVN learning algorithm is based on the error-correction rule and it works as follows. Let D and Y be, respectively, the desired and actual outputs of the continuous MVN. Then the weight adjustment formula is:

$$W_{r+1} = W_r + \frac{C_r}{(n+1)|z_r|}(D-Y)\overline{X} \tag{2}$$

where $\overline{X}$ is the neuron input vector with the complex-conjugate components, r is the index of the learning iteration, n is the number of neuron inputs (dimension of the input vector), W_r and W_{r+1} are the weighting vectors (complex), before (current) and after correction, respectively, C_r is the learning rate, empirically chosen around 1.

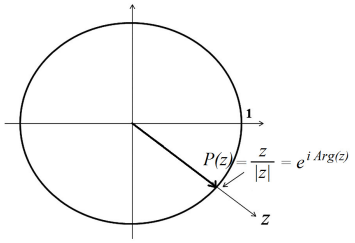

Fig. 1: Geometrical interpretation of the continuous MVN activation function.

Geometrically, (2) represents the movement along the unit circle in the shortest possible way from the "incorrect" actual output to the "correct" desired output (see Fig. 2). The training of the neuron using this rule is performed without requiring any derivative of the activation function and the convergence of the corresponding learning algorithm is proven in [12].

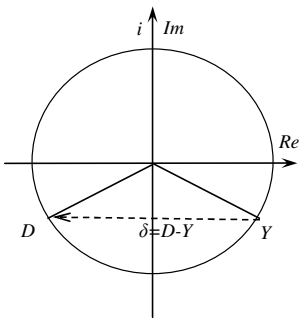

Fig. 2: Geometrical interpretation of the MVN learning rule.

The MLMVN has a standard feedforward topology, where the inputs of all the neurons of a layer are connected to the corresponding outputs of the neurons of the previous layer. However, the use of MVN as basic neuron for the MLMVN determines some important characteristics and advantages of this network with respect to the standard multi-layer backpropagation perceptron based network (MLBPN). On the other hand, also in this kind of neural network a backpropagation algorithm is used to correct the weights during the training phase. How the learning algorithm is organized for an MLMVN with m layers is shown in [13]. For the output layer (if we have m layers, it is the mth layer) the weights for the kth neuron of this (mth) layer have to be adjusted according to the following rule:

$$\begin{aligned} \tilde{w}_i^{km} &= w_i^{km} + \frac{C_{km}}{(N_{m-1}+1)}\delta_{km}\overline{\tilde{Y}}_{i,m-1} \quad i = 1,\ldots,N_{m-1} \\ \tilde{w}_0^{km} &= w_0^{km} + \frac{C_{km}}{(N_{m-1}+1)}\delta_{km} \end{aligned} \tag{3}$$

where $\tilde{w}$ represents the corrected weight, N_{m-1} is the number of neurons in the $(m-1)$th layer (the last hidden layer preceding to the output one) corresponding also to the number of inputs of all neurons in the mth layer, C_{km} is the learning rate (it should always be equal to 1), $Y_{i,m-1}$ is the actual output of the ith neuron of the $(m-1)$th layer (it is intended corrected when has the ˜ superscript and conjugated when has the "bar" superscript), and $\delta_{km} = \frac{1}{(N_{m-1}+1)}\delta_{km}^*$ is the error of the kth neuron of the mth (output) layer, which is obtained from the global network error taken from the same neuron:

$$\delta_{km}^* = D_{km} - Y_{km} \tag{4}$$

where D_{km} and Y_{km} are, respectively, the desired and actual outputs of the kth neuron of the mth layer.

For the hidden layers neurons, except the first one, the error is calculated using its backpropagation from the following layer and then the weights should be adjusted as follows:

$$\begin{aligned} \delta_{kj} &= \frac{1}{(N_{j-1}+1)}\sum_{i=1}^{N_{j+1}}\delta_{ij+1}(w_k^{ij+1})^{-1} \\ \tilde{w}_i^{kj} &= w_i^{kj} + \frac{C_{kj}}{(N_{j-1}+1)|z_{kj}|}\delta_{kj}\overline{\tilde{Y}}_{i,j-1} \quad i = 1,\ldots,N_{j-1} \\ \tilde{w}_0^{kj} &= w_0^{kj} + \frac{C_{kj}}{(N_{j-1}+1)|z_{kj}|}\delta_{kj} \end{aligned} \tag{5}$$

where the indexes kj stand for the kth neuron of the jth layer, whose weights are adjusted, $|z_{kj}|$ is the absolute value of the current weighted sum of this neuron, N_{j-1} is the number of neurons in the $(j-1)$th layer (this is also the number of inputs of all neurons in the jth layer), C_{kj} is the learning rate (it should always be equal to 1), $Y_{i,j-1}$ is the actual output of the ith neuron of the $(j-1)$th layer (to be intended corrected when has the ˜ superscript and conjugated when has the "bar" superscript), δ_{kj} is the error of the kth neuron of the jth layer. Finally, for the first hidden layer the error is backpropagated from the second hidden layer and then the weights should be adjusted as follows:

$$\begin{aligned} \delta_{k1} &= \frac{1}{s_j}\sum_{i=1}^{N_2}\delta_{i2}(w_k^{i2})^{-1} \\ \tilde{w}_i^{k1} &= w_i^{k1} + \frac{C_{k1}}{(n+1)|z_{k1}|}\delta_{k1}\overline{x}_i \quad i = 1,\ldots,n \\ \tilde{w}_0^{k1} &= w_0^{k1} + \frac{C_{kj}}{(n+1)|z_{k1}|}\delta_{k1} \end{aligned} \tag{6}$$

where the index $k1$ stands for the kth neuron of the 1st layer, $\overline{x}_i$ is the complex conjugate of the ith input component, n is the number of network inputs (it is also the number of inputs

of all the 1st hidden layer neurons), C_{k1} is the learning rate, and δ_{k1} is the error of the kth neuron of the 1st layer. The convergence of the learning process based on the learning rules (3)-(6) is proven in [12] and [13] for the MLMVN with a single output neuron and with multiple output neurons, respectively.

As mentioned above, the main advantages of the MLMVN are its higher functionality comparing to the MLBPN, the simplicity of its learning, which is derivative-free and does not depend on the choice of a learning rate (it is self-adaptive), and better performance in terms of classification/prediction rate [12-15].

B. Modification of the MLMVN Learning Algorithm Using Complex QR Decomposition

Despite the very good results of the described learning algorithm, it was noticed that convergence usually requires too many learning iterations. In order to reduce this time, an important modification of the MLMVN learning algorithm has been introduced by the authors, which ensures a faster convergence. The new approach is applied to the calculation of the errors of the output layer neurons. The modified learning procedure minimizes the learning error of every training epoch by means of the LLS (Linear Least Squares) algorithm which uses the complex QR decomposition. This modification significantly improves the MLMVN performance ensuring much faster convergence of the learning process. In order to achieve it we may consider the learning process not over the complete output, but trying to adapt in the right direction the weighted sum of the last layer inputs. To be more specific, we may estimate the desired value of the weighted sum by the desired output value. Let us consider, for example, the MVN learning rule (2). Let n be the number of neuron inputs, D be the desired output, z be the current value of the weighted sum. Then the error is the following:

$$\delta = D - z \tag{7}$$

Starting by this new definition of the error, the weights can be adjusted according to (2). Let $\tilde{z}$ be the adjusted value of the weighted sum and $\tilde{w}_0, \tilde{w}_1, ..., \tilde{w}_n$ be the adjusted weights. Let us find $\tilde{z}$ (from here let $C_r = 1$ for simplicity):

$$\begin{aligned} &\tilde{z} = \tilde{w}_0 + \tilde{w}_1 x_1 + ... + \tilde{w}_n x_n = \\ &= (w_0 + \frac{\delta}{(n+1)}) + (w_1 + \frac{\delta}{(n+1)}\overline{x}_1)x_1 + ... + (w_n + \frac{\delta}{(n+1)}\overline{x}_n)x_n = \\ &= w_0 + \frac{\delta}{(n+1)} + w_1 x_1 + \frac{\delta}{(n+1)} + ... + w_n x_n + \frac{\delta}{(n+1)} = \\ &= w_0 + w_1 x_1 + ... + w_n x_n + \delta = z + \delta = D \end{aligned} \tag{8}$$

Thus, after the adjustment of the weights, the weighted sum is exactly equal to the desired output. This change of the error definition is illustrated in Fig. 3.

It is possible to demonstrate [16] that, if our learning set contain M samples, and δ_j is the learning error for the considered sample, then the following system of linear algebraic equations is generated over the field of complex numbers, with unknowns $\Delta w_0, \Delta w_1, ..., \Delta w_n$:

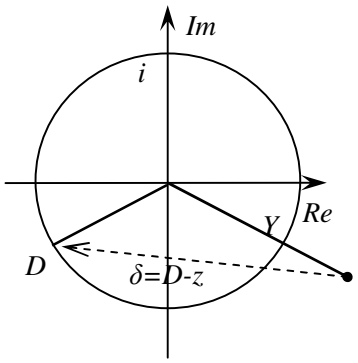

Fig. 3. Geometrical interpretation of the new error definition.

$$\begin{cases} \Delta w_0 + x_1^1 \Delta w_1 + ... + x_n^1 \Delta w_n = \delta_1 \\ \Delta w_0 + x_1^2 \Delta w_1 + ... + x_n^2 \Delta w_n = \delta_2 \\ \quad ... \quad ... \quad ... \quad ... \quad ... \quad ... \\ \Delta w_0 + x_1^M \Delta w_1 + ... + x_n^M \Delta w_n = \delta_M \end{cases} \tag{9}$$

The system (9) is constituted by M equations with n+1 unknowns. This system is typically overdetermined because $M >> n$+1 for most of the practical problems (as well as for most of the benchmarks). As well known (see for example [17]), using the LLS algorithm, it is possible to find the solution which minimizes the sum of the squared of the following terms:

$$\delta_j - \left(\Delta w_0 + x_1^j \Delta w_1 + ... + x_n^j \Delta w_n\right) \quad j = 1, ..., M$$

This algorithm can be efficiently implemented, for example, by means of the QR decomposition [18].

The QR decomposition is a classical method for the factorization of a complex-valued $m \times n$ matrix A, with $m \geq n$. It consists in the decomposition of a matrix as product of an $m \times m$ unitary matrix Q and an $m \times n$ upper triangular matrix R [18]:

$$\mathbf{A = QR}$$

In our case we have to decompose the following $M \times (n+1)$ complex matrix:

$$\mathbf{A} = \begin{pmatrix} 1 & x_1^1 & ... & x_n^1 \\ 1 & x_1^2 & ... & x_n^2 \\ ... & ... & ... & ... \\ 1 & x_1^M & & x_n^M \end{pmatrix} \tag{10}$$

The QR decomposition generates an $M \times M$ unitary matrix $\mathbf{Q}$ and an upper triangular matrix $\mathbf{R}$ of the following kind:

$$\mathbf{R} = \begin{pmatrix} \mathbf{R}_{n+1} \\ 0 \end{pmatrix}$$

where $\mathbf{R}_{n+1}$ is an $(n+1) \times (n+1)$ upper triangular matrix.

The QR decomposition of a matrix always exists [18], even if the matrix does not have a full rank. To find the QR decomposition, the Householder transformation (called also Householder reflection) [19] can be used. This brings to the final relation:

$$\mathbf{R}\cdot\Delta\mathbf{w}^T=\mathbf{Q}^{-1}\boldsymbol{\delta}^T=\mathbf{Q}^T\boldsymbol{\delta}^T$$

from which $\Delta\mathbf{w}$ can be easily found, since the matrix $\mathbf{R}$ is an upper triangular matrix.

The computational complexity for the QR decomposition by Householder technique is $\mathrm{O}\left(M(n+1)^2-\frac{1}{3}(n+1)^3\right)$ [20]. The computational complexity of each step of the MLMVN algorithm can be calculated combining the various products. N_x is the number of inputs of the network. For each of the M examples the following operations are repeated (in the most common case of one hidden layer):

- $n(N_x+1)+(n+1)$ operations for the output calculation;
- $2 + 2n$ operations for the backpropagation of the error;
- $n[(N_x+1)+(N_x+1)]$ operations for the hidden layer weight adjustment (calculation and adjustment);
- $n(n+1)$ operations for the output layer weight adjustment.

Then, adding up these terms, the computational complexity of the classical MLMVN algorithm can be finally determined, resulting of the order:

$$\mathrm{O}\left(3M\cdot n(N_x+1)+4M(n+1)\right).$$

Recalling that, usually, $M >> n+1$, and N_x and n can reasonably be considered of the same order of magnitude, the extra complexity introduced by the QR decomposition is acceptable (due to the dramatic reduction of the number of training epochs, see Table I). If there are multiple output neurons in the network, the same procedure has to be separately applied to each output neuron. This new approach for the error calculation and weights adjustment must be used only for the output layer neurons, because in this case the exact errors can always be easily calculated. On the contrary, since for the hidden layer neurons the exact errors are never known and they are obtained using the backpropagation technique, the learning procedure described in section A must be used instead.

To summarize, the proposed modified MLMVN learning algorithm consists of the following steps:

1. *for each neuron of the output layer, the inputs are stored in a complex $M\times(n+1)$ matrix A and the output errors are stored in a complex M-dimensional vector $\boldsymbol{\delta}$; at the end of the learning epoch, the adjustments for the weights of the output neurons, Δw_i, i = 0,1, …, n (n is the number of neuron inputs) are computed using the complex QR decomposition, applied to the overdetermined system of linear algebraic equations $\mathbf{A}\cdot\Delta\mathbf{w}^T=\boldsymbol{\delta}^T$;*
2. *for the hidden layers neurons, the error is backpropagated and the weights are adjusted according to (5) for all the hidden layers except the first one, and according to (6) for the first hidden layer. The backpropagation of the error to each layer, except the output one, is done for each learning sample separately. The adjustments calculated at every step are kept and used in the following construction of the QR system and for the correction of the weights of the output neurons.*

The proposed modified learning algorithm gives noteworthy advantages in terms of learning speed, that is in terms of number of learning epochs required to obtain the desired error, while preserving the performance advantages of the MLMVN (or even slightly enhancing them) in terms of generalization capability. A complete comparison among the new version of the multi-valued neural network, the classical version, and other traditional NNs for classical and original benchmarks, can be found in [16]. In Table I a short extract of the comparison is reported.

TABLE I. COMPARISON TABLE FOR MACKEY-GLASS BENCHMARK

Type of network and learning algorithm	N. of neurons in the hidden layer	Min. test RMSE	N. of training epochs for the min. test RMSE	Average test RMSE	Average number of training epochs	Av. Exec. time in sec (on a PC Dual Core 2.20 GHz)
MLMVN-QR	50	0.0065	35	0.0068	50	12
MLMVN	50	0.0056	95,381	0.0066	162,18	8640
MLBPN	50	0.0212	23,800	0.0235	27150	980

In Table I it can be seen that the canonical version of MLMVN outperforms the classical backpropagation in terms of error, but it needs many more thousands of epochs with respect to the new QR-modified version, which has the same error magnitude of the canonical one.

III. NEURAL ARCHITECTURE FOR THE PARAMETER IDENTIFICATION

In the previous section the better performance of the proposed network with respect to the other inversion paradigms (as MLBPN), in term of both epochs required for the training and number of required neurons, has been verified. Now it is used in a double configuration with the aim to find an approximate relation between the geometrical features of a distributed or complex system and its lumped model in all the cases in which an exact relation cannot be easily found. The relation can be expressed as $\mathbf{p} = f(\mathbf{g})$, where $\mathbf{p}$ is the lumped model parameter vector and $\mathbf{g}$ is the geometrical parameter vector. We assume that in a generic distributed device or apparatus, the frequency response in a given significant range is dependent on the geometrical attributes of the device and that we can measure or, more usually, simulate this dependence (for instance via a sophisticated FEM simulator or by solving a set of differential equations). Then, if $\mathbf{H}=h(\mathbf{p},\omega)=h(f(\mathbf{g}),\omega)$, where $\mathbf{H}$ is the response vector in the frequency domain (or the FFT of a time domain response), it can be assumed that the geometrical parameters are available and the frequency or time domain response can be measured or simulated for that set of geometrical parameters over a suitably chosen equivalent lumped model.

In Fig. 4 the general scheme is shown (DMLMVN means Double MLMVN). The MLMVN1 neural network is used for approximating the relationship between geometrical

parameters and frequency response; it is trained over the set {**g**, **H**} during the learning phase, and it generates the estimated response **H*** during the work phase, separating magnitude and phase response (in other words, one neuron of the output layer is used for the magnitude and another one for the phase). Since the desired output is the equivalent model, we have to determine, basically, the relationship between the lumped parameters **p** and the geometrical parameters **g**, $\mathbf{p} = f(\mathbf{g})$. The problem is that the supervised signal for that inversion is unknown, because the lumped model must still be validated (i.e. associated to the right component values). In order to fill the gap, another MLMVN is included in the system (the part MLMVN2), whose aim is the inversion of the frequency response $\mathbf{p} = h^{-1}(\boldsymbol{H})$. This network is trained over a set of data prepared using the lumped model of the circuit to be identified, collected (simulated) in a significant range of frequencies (the frequency range where the circuit operates). Hence the outputs of this stage consist of a numerical estimation **p*** of the electric features of the device under analysis, after the learning phase performed over the set {**H**, **p**}.

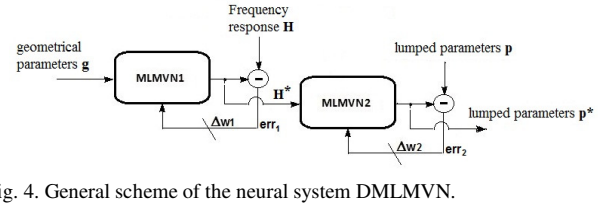

Fig. 4. General scheme of the neural system DMLMVN.

For the second neural network the measurements corresponding to the values of the network function in a selected set of frequencies are the inputs and the parameter values of the lumped circuital model of the device are the outputs. Then the task of the second neural network is the parameter extraction of the lumped model starting from the network function measurements. In order to suitably design the architecture of the network, it is necessary to determine a priori the solvability degree of our problem. To this end let us consider a network function of a lumped circuit expressed in the following way:

$$H(\mathbf{p},\omega)=H(\mathbf{p},s)\Big|_{s=j\omega}=\frac{N(\mathbf{p},s)}{D(\mathbf{p},s)}\Big|_{s=j\omega}=\left.\frac{\sum_{i=0}^{n}\frac{a_i(\mathbf{p})}{b_m(\mathbf{p})}\cdot s^i}{s^m+\sum_{j=0}^{m-1}\frac{b_j(\mathbf{p})}{b_m(\mathbf{p})}\cdot s^j}\right|_{s=j\omega} \tag{11}$$

where $\mathbf{p}=[p_1,\ p_2,\ .....p_R]^t$ is the vector of the circuit parameters. Starting from a set of measurements corresponding to the values of the network function in a selected set of frequencies, it is possible to determine the coefficients of the network function, for example by means of a LSE (Least Squared Error) procedure [21]. At this point a nonlinear system expressing these coefficients as functions of the unknown circuit parameters can be considered. In order to determine the solvability degree of this system it is sufficient to evaluate the rank of the Jacobian matrix **B** associated to the system, where each column is related to a specific circuit parameter:

$$\mathbf{B}=\begin{bmatrix} \frac{\partial\frac{a_0}{b_m}}{\partial p_1} & \frac{\partial\frac{a_0}{b_m}}{\partial p_2} & \cdots & \frac{\partial\frac{a_0}{b_m}}{\partial p_R} \\ \cdots & \cdots & & \cdots \\ \frac{\partial\frac{a_n}{b_m}}{\partial p_1} & \frac{\partial\frac{a_n}{b_m}}{\partial p_2} & \cdots & \frac{\partial\frac{a_n}{b_m}}{\partial p_R} \\ \cdots & \cdots & & \cdots \end{bmatrix} \tag{12}$$

If rank **B** is equal to the number of unknown parameters R, their values can be uniquely determined, if it is less than R, a locally unique solution can be determined only if R – rank **B** parameters are fixed a priori.

This kind of approach is the same followed in the parametric fault diagnosis problems of analog circuits, where rank **B** is named testability [22-24]. Then it is possible to exploit all the theoretical results obtained in this field. In particular, in fault diagnosis field, each set of linearly dependent columns of **B** locates an ambiguity group constituted by the circuit parameters corresponding to these columns [23]. An ambiguity group is, essentially, a group of parameters where it is not possible to uniquely identify the value of each of them starting from the measurement data. The knowledge of the ambiguity groups allows us to choose the set of parameters to consider as unknowns, named testable group. As demonstrated in [25], both testability value and ambiguity groups do not depend on component values, then they can be determined by assigning arbitrary values to the parameters.

In our case the lumped model of the device is chosen a priori, then the network function is determinable in symbolic form with respect to the unknown parameter values. No system has to be solved, only the matrix in (12) has to be considered in order to determine its rank and the sets of linearly dependent columns by using arbitrary values for the parameters. The rank gives us the solvability degree of the problem, that is it indicates how many parameters must be fixed a priori, the sets of linearly dependent columns give us the ambiguity groups, that allow us to determine the testable group, that is the parameters to consider as unknowns in order to obtain a unique solution. Furthermore, being the solvability degree (testability) linked with the selected network function, this information can be used to find the network function of the model giving the maximum solvability. Once the network function and the unknown parameters have been chosen, the unknown parameters become the outputs of the second neural network. The measurements are carried out and the learning process is performed. If the learning process converges, the validation of the lumped model is obtained and the circuit parameter extraction is achieved. The parameter extraction procedure can be summarised as follows:

1. *establish the lumped model equivalent circuit of the device;*

2. *calculate the solvability degree (testability) and evaluate which parameters can be assumed as unknowns;*
3. *generate an adequate number of samples to be used in the training phase;*
4. *train the NN_2 part of the network illustrated in Fig. 4;*
5. *train the overall network illustrated in Fig. 4;*
6. *extract the parameters, as the output of the NN_2 part of the whole system.*

It is worth pointing out that the solvability degree evaluation and ambiguity group determination can be very efficiently performed by means of symbolic techniques [24]. The symbolic algorithms used for this task [24] have low computational complexity and do not significantly increase the computational cost of the whole parameter extraction system.

IV. Examples

In this section two examples are used to illustrate the method for extracting the parameters of a lumped model from a set of measurements or simulations. It should be underlined that the data can come from measurements, simulations, or also from analytical formulas or equations that are approximated by the overall neural system.

A. Lumped model of a winding inductor

As a first example, let us consider the identification of the stray capacitance C_s of a winding inductor [26]. It is known that a winding inductor is a system that can be described with a lumped model based on the geometrical characteristics of the coil and of the conductors. In particular, inductor windings have a distributed parasitic capacitance, that can be modeled by a lumped capacitance connected between the terminals of the winding, as shown in Fig. 5, together with a cross-sectional view of a winding.

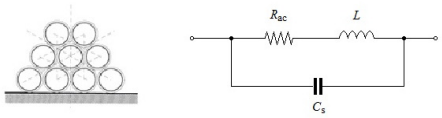

Fig. 5: Section representation of the winding inductor and electric scheme.

The evaluated function is the impedance of the inductor. A set of examples is generated with SPICE simulator, constituted of 250 different combinations of the three geometric parameters given by the diameter of the coil turn d_t, the inner diameter (D_c) and the outer diameter (D_o) of the conductor wire. The dielectric constant of the coating material used for the nonimpregnated coil is ε_r = 3.5. For every combination of these parameters the impedance is sampled with 200 frequency points, in the range 1-100 MHz, which includes the resonance frequency of the inductor. As a result, the input pattern is constituted of a matrix with 250 rows and 3 columns and the output pattern is a matrix of 250 rows and 400 columns (magnitude and phase of the frequency response). In this case, the learning process successfully recognizes the parameter C_s, once its distinguishability has been verified through the determination of testability and ambiguity groups. The fixed parameter L is set to the design value L= 75 μH, the parasitic resistance R_{ac} is calculated via analytical formulas [26]. Table II shows a comparison among some values of stray capacitance extracted by the network and analytically evaluated for a given configuration.

TABLE II. Stray Capacitance values

Capacitance true value, nF	*Capacitance value from NN2, nF*	*Error*	*N° of epochs*
483.98	481.39	0. 54%	16
695.76	700.96	0.747%	16
472.27	461.84	2.21%	16

The final error is kept small in every example, as it is evident, and it is calculated comparing the output of the neural system with the one coming from analytical expressions.

B. Lumped model of a microwave BP filter

As second example, let us consider the lumped parameter model of a two-post microwave filter, like that represented in Fig. 6. The related equivalent circuit is shown in Fig. 7. The filter is a bandpass with a narrow band. In particular, it has a center frequency in the GHz range and a bandwidth in the MHz range. Furthermore, as outlined in [27], a very small variation in the geometric dimensions of the filter causes a consistent variation in the position of the center frequency. For this reason, in order to obtain a sufficient definition of the filter response, it is necessary to sample it with a very small frequency step. To this purpose, a uniform sampling is used, with 1500 samples over a 12.5 to 13.5 GHz band.

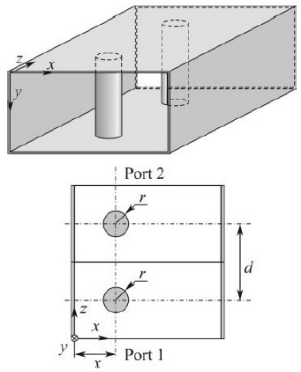

Fig. 6: Geometry of the two-post waveguide filter.

The set of examples is composed of 1727 different geometric combinations on this interval corresponding to 1727 different filter behaviours. A hybrid finite-element–modal-expansion (FEM-ME) technique is used for the computation of the Generalized Scattering Matrix (GSM) of the device [28]. The geometry of the filter is completely represented by the three geometrical parameters indicated in Fig. 6: *x*, *r* and *d*. The equivalent lumped model is derived from the filter response, which is well represented by a 4^{th} order transfer function. The determination of testability and ambiguity groups of the model is in this case more complicated; the lumped model filter, in fact, has a testability

value equal to 6, 272 ambiguity groups and 1306 testable groups. The model identification has been made by choosing a set of unknown parameters constituted by the components R_1, R_6, R_9, C_1, C_2, C_3, and training the NN_2 over a combination of 4000 variations of these parameters. After the completion of the learning phase of the inversion module NN_2, the overall network is learned over the first set of samples in order to give as output the electrical lumped parameters R_1, R_6, R_9, C_1, C_2, C_3. In Fig. 8, a single output is reported in the frequency domain, relating to the geometrical

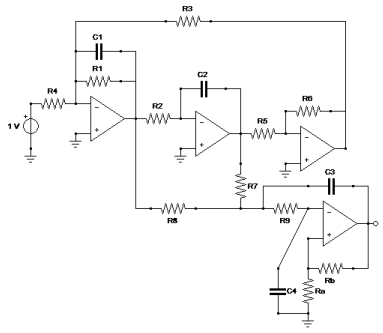

Fig. 7: Equivalent lumped circuit of the microwave filter.

parameter set: {x, r, d} = {8.525, 2.743, 18.11} mm. The training is carried out in about 50 epochs for the module 2 and 35 for the module 1 (and final result) and it gives the following six parameter values: R_1=208.31 Ω, R_6=0.98706 Ω, R_9=0.21416 Ω, C_1=0.9013 pF, C_2=1.143 pF, C_3=0.9879 pF.

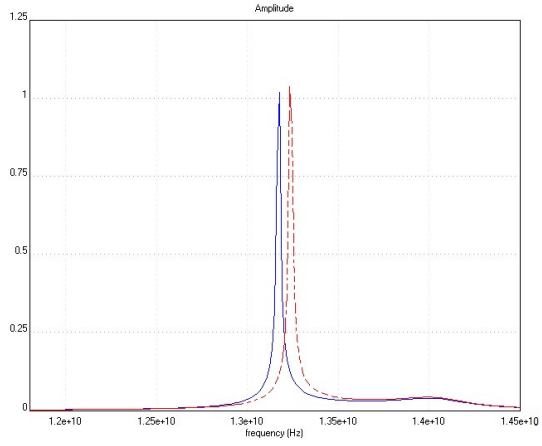

Fig. 8: Frequency response amplitude of the microwave circuit simulated by lumped parameter model (dashed curve) compared with the one generated by FEM simulator (solid curve).

In this second example the comparison has been made directly over the output frequency response (amplitude response), because the relationship between geometrical parameters and frequency response is calculated via FEM analysis and no analytical formula is directly available, differently from the previous case. The error over the central frequency is 2.92%, that over the amplitude is 2.67%.

V. Conclusion

A neural system, based on a pair of complex-valued neural networks, has been presented in this paper. It performs the identification of a lumped model associated to an electrical device by extracting the relations between the geometrical features of the device and the electrical parameters of the lumped model, using also the frequency response. The accuracy level of the achieved results for some significant cases proposes it as a valid tool for the modelization of distributed devices. Future work will be devoted to a more accurate *a-priori* evaluation of the cumulated error.

References

[1] Q. J. Zhang, L. Ton, and Y. Cao, "Microwave modeling using artificial neural networks and applications to embedded passive modeling," in *Proc. of Microwave and Millimeter Wave Technology Conf.(ICMMT 2007),* Guilin, 2007, pp. 1-4.

[2] G. Avitabile, B. Chellini, G. Fedi, A. Luchetta, and S. Manetti, "A neural architecture for the parameter extraction of high frequency devices," in *Proc. of IEEE Int. Symposium on Circuits and Systems (ISCAS2001)*, Sidney, 2001, pp. 577-580.

[3] R. Pratap, D. Staiculescu, S. Pinel, J. Laskar, and G. May, "Modeling and sensitivity analysis of circuit parameters for flip-chip interconnects using neural networks," *IEEE Trans. on Advanced Packaging,* vol. 28, pp. 71-78, Feb. 2005.

[4] A. Shinterimov, W. H. Tang, and Q. H. Wu, "Transformer core parameter identification using frequency response analysis", *IEEE Trans. on Magnetics*, vol. 46, pp. 141-149, Jan. 2010.

[5] V. Rashtchi, E. Rahimpour, and E. M. Rezapour, "Using a genetic algorithm for parameter identification of transformer R-L-C-M model," *Electrical Engineering,* vol. 88, pp. 417-422, June 2006.

[6] A. Shinterimov, W. J. Tang, W. H. Tang, and Q. H. Wu, "Improved modelling of power transformer winding using bacterial swarming algorithm and frequency response analysis," *Electric Power Systems Research*, vol. 80, pp. 1111-1120, Sep. 2010.

[7] W. H. Tang, S. He, Q. H. Wu, and Z. J. Richardson, "Winding deformation identification using a particle swarm optimiser with passive congregation of power transformes," *Int. J. of Innovations in Energy Systems and Power,* vol. 1, pp. 46-52, Nov. 2006.

[8] I. Aizenberg, N. N. Aizenberg, and J. Vandewalle, *Multi-valued and universal binary neurons, theory, learning and applications*. Dordecht: Kluwer Academic Publishers, 2000.

[9] G. Fedi, A. Luchetta, S. Manetti, and M. C. Piccirilli, "A new symbolic method for analog circuit testability evaluation," *IEEE Transactions on Instrumentation and Measurement*, vol. 47, pp. 554-565, April 1998.

[10] N. N. Aizenberg, and Y. L. Ivaskiv, *Multiple-valued threshold logic*. Kiev: Naukova Dumka Publisher House (in Russian), 1977.

[11] I. Aizenberg, *Complex-Valued Neural Networks with Multi-valued Neurons*. Berlin: Springer Verlag, 2011.

[12] I. Aizenberg, D. Paliy, J. Zurada, and J. Astola, "Blur identification by multilayer neural network based on multivalued neurons," *IEEE Transactions on Neural Networks,* vol. 19, pp. 883-898, May 2008.

[13] I. Aizenberg, and C. Moraga, "Multilayer feedforward neural network based on multi-valued neurons (MLMVN) and a backpropagation learning algorithm," *Soft Computing*, vol. 11, pp. 169-183, Jan. 2007.

[14] I. Aizenberg, and C. Moraga, "The genetic code as a function of multiple-valued logic over the field of complex numbers and its learning using multilayer neural network based on multi-valued neurons," *Journal of Multiple-Valued Logic and Soft Computing*, vol. 4-6, pp. 605-618, Nov. 2007.

[15] I. Aizenberg, and J. Zurada, "Solving selected classification problems in bioinformatics using multilayer neural network based on multi-valued neurons (MLMVN)," in 2007 *Proc. of the International Conference on Artificial Neural Networks (ICANN-2007),* Porto.

[16] I. Aizenberg, A. Luchetta, and S. Manetti, “A modified learning algorithm for the multilayer neural network with multi-valued neurons based on the complex QR decomposition,” *Soft Computing*, vol. 16, pp. 263-275, April 2012.

[17] C. L. Lawson, and R. J. Hanson, *Solving least squares problems*. Englewood Cliffs: Prentice-Hall, N.J, 1974.

[18] G. H. Golub, and C. F. Van Loan, *Matrix computations* (3rd ed.), Baltimore: Johns Hopkins University Press, 1996.

[19] A. S. Householder, “Unitary triangularization of a nonsymmetric matrix,” *Journal of the ACM*, vol. 5, pp. 339–342, October 1958.

[20] J. Stoer, and R. Bulirsch, *Introduction to numerical analysys.* New York: Springer-Verlag Publishers, II edition, 1991.

[21] G. Fedi, R.Giomi, A. Luchetta, S. Manetti,and M. C. Piccirilli, “On the application of symbolic techniques to the multiple fault location in low testability analog circuits,” *IEEE Transactions on Cicuits and Systems II*, vol. 45, pp. 1383-1388, October 1998.

[22] A. Liberatore, S. Manetti, and M. C. Piccirilli, “A new efficient method for analog circuit testability measurement,”in *Proc. of IMTC’94*, Hamamatsu, 1994, pp. 193-196.

[23] G. Fedi, S. Manetti, M. C. Piccirilli, and J. Starzyk, “Determination of an optimum set of testable components in the fault diagnosis of analog linear circuits,” *IEEE Transactions on Circuits and Systems - Part I*, vol. 46, pp. 779-787, July 1999.

[24] S. Manetti, and M. C. Piccirilli, “A singular-value decomposition approach for ambiguity group determination in analog circuits,” *IEEE Transactions on Circuits and Systems – Part I*, vol. 50, pp. 477-487, April 2003.

[25] N. Sen, and R. Saeks,“Fault diagnosis for linear systems via multifrequency measurement,” *IEEE Trans. Circ. and Syst.*, vol. 26, pp. 457-465, July 1979.

[26] M. Bartoli, A. Reatti, and M. K. Kazimierczuk, “Modeling of ironpowder inductors at high frequencies,” in *Proc. IEEE Industry Applications Conf.,* Denver, 1994, pp. 1225–1232.

[27] G. Fedi, S. Manetti, G. Pelosi, and S. Selleri, “FEMtrained artificial neural networks for the analysis and design of cylindrical posts in rectangular waveguide,” *Electromagnetics,* vol. 22, pp. 323–330, April 2002.

[28] G. Pelosi, R. Coccioli, and S. Selleri, *Quick finite elements method for electromagnetic waves*. London: Artech House, 1998, pp. 89-113.