IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 25 November 2022, accepted 29 November 2022, date of publication 1 December 2022,
date of current version 12 December 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3226323

RESEARCH ARTICLE

A Hybrid Approach Based on Recurrent Neural Network for Macromodeling of Nonlinear Electronic Circuits

AMIN FARAJI[1], SAYED ALIREZA SADROSSADAT[1], (Member, IEEE), MAHDI YAZDIAN-DEHKORDI[1], MORTEZA NABAVI[2], (Member, IEEE), AND YVON SAVARIA[2], (Fellow, IEEE)

[1]Department of Computer Engineering, Yazd University, Yazd 8915818411, Iran
[2]Department of Electrical Engineering, Polytechnique Montreal, Montreal, QC H3T 1J4, Canada

Corresponding author: Sayed Alireza Sadrossadat (alireza.sadr@yazd.ac.ir)

ABSTRACT This paper proposes a hybrid approach combining Recurrent Neural Network (RNN) and polynomial regression methods for time-domain modeling of nonlinear circuits. The proposed hybrid RNN-polynomial regression (HRPR) method merges RNN and polynomial regression which leads to a significant reduction in training time while providing speedup in simulation compared to both conventional RNN and existing models in simulation tools without sacrificing accuracy. The proposed HRPR method comprises two steps: First, an RNN structure is generated, and then, the output of the RNN is combined with external input(s) of the circuit to perform a regression. Applying this method causes part of the training process to be done by polynomial regression which is simpler than training an RNN. Also, the RNN used in the HRPR method has a simpler structure than a single conventional RNN used for modeling the same component. To verify the validity of the proposed method, modeling and comparisons of three nonlinear examples are presented in this paper.

INDEX TERMS Computer-aided design (CAD), recurrent neural network, polynomial regression, nonlinear components modeling, simulation.

I. INTRODUCTION

With the increasing complexity of systems exploiting nonlinear circuits and components, control and macromodeling them with high accuracy remains an important concern and active research area [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12]. Macromodeling is an approach for creating efficient circuit models that reduces the amount of information required to handle them. In other words, a macromodel can be viewed as a compact abstraction of a circuit. With macromodeling, only information necessary to calculate some desired output variables is retained, while the rest of the data can be suppressed [1]. Several macromodeling approaches have been used in the literature, such as inertial delayed Elmore delay (DED) [1], Trajectory PieceWise (TPW) method [13], Recursive Vector Fitting (RVF) [14], and the Neuro-Space Mapping approach [15].

With the development of new technologies, existing models may become insufficiently accurate. Thus, existing models may need to be modified or improved [16], [17]. However, developing a new equivalent circuit model, which usually requires manual trial-and-error efforts, is a time-consuming procedure. As an alternative approach, artificial neural networks (ANNs) have been introduced in the literature for nonlinear device control and modeling and have contributed to the evolution of computer-aided design (CAD) of circuits and systems [2], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27] ANN-based CAD methods have led to a notable improvement in efficiency and speed of modeling. Optimization and simulation of circuits and components have been frequently used for modeling of nonlinear circuits and systems [28], [29], [30], [31]. ANNs can learn input-output

The associate editor coordinating the review of this manuscript and approving it for publication was Mu-Yen Chen.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

relationships. Trained model can be used in circuit simulators to provide quick and accurate responses [32]. They can be developed from external signals of the original circuit without implementing its internal details. This capability is very useful for modeling a new device or circuit when its analytical representation is not available, or when a detailed model is too computationally expensive to evaluate [32]. Several time-dependent neural networks such as dynamic neural networks (DNNs) [33], time-delay neural networks (TDNNs) [34], echo state network [35], Long Short-Term Memory (LSTM) [36], recurrent neural networks (RNNs) [32], [37], [38], [39], [40], [41], [42], [43], [44], [45] and state-space dynamic neural network [29], [46], and recently batch normalized recurrent neural network [47] have been proposed in the literature in order to obtain high-performance models for nonlinear circuits and components. Also, there are other modeling methods such as fractional order methods which rely on solving differential equations. Fractional order methods use fewer parameters, reduce complexity, and are simpler but neural network use more parameters and more complex to train but lead to more accuracy. Also, using differential equations for system identification results in more sensitivity to noisy data where discrete-time RNNs do not have that challenge. Also, models based on fractional orders usually can be used for steady-state response of circuits but the RNN-based model can capture both transient and steady-state behavior of nonlinear circuits accurately [48], [49], [50], [51]

Among all these methods, RNNs are broadly used for macromodeling the time-domain response of nonlinear circuits. Their capability is notably due to the universal approximation aspect of RNNs, which can be trained to learn and approximate virtually any sophisticated input-output relationship [38]. RNNs have many parameters that can be trained. Also, recursive training through multiple steps and layers makes RNN training a time-consuming technique.

In this paper, a method that combines RNN with polynomial regression has been introduced to solve these issues. For the first time in modeling nonlinear circuits and components, regression units are added on top of the RNN to perform as a part of the model training. This hybrid method results in a significant reduction in the number of parameters compared to the conventional RNN method. The reason is that the RNN which is used prior to the regression units, has a smaller structure compared to the one (without regression) which is required for modeling the same circuits with suitable accuracy. The goal of regression is to predict the values of one or more continuous target variables given the values of a multi- dimensional vector x of input variables [52]. There are several types of regression methods, such as linear regression and polynomial regression. Regression has been used in different areas and applications, such as face recognition [53], price forecasting [54], decoding muscle activation pattern [55], and estimation of human affective states [56]. Our proposed hybrid method presented in this paper uses polynomial regression as part of the training process. Thus, the training time for modeling nonlinear components is significantly reduced. This reduction in training time is in such a way that the model obtained from the proposed method is trained multiple times faster than the conventional methods. Other than reducing training time, this method reduces test time (evaluation of response) compared to conventional RNN and existing circuit simulation tools [57]. This is due to using fewer number of parameters in the proposed method. Indeed, some of the inputs of the adopted regressions can be obtained from simple RNNs containing few parameters, as compared to conventional RNN methods which contain many parameters to be trained. Therefore, the method proposed in this paper can outperform the conventional RNN technique in both speed and accuracy.

This paper is organized as follows: the conventional RNN structure and its use for macromodel development are presented in section II. The proposed HRPR method is presented in section III. Validation of this method using three examples is reported in section IV. Finally, conclusions drawn from this research are presented in section V.

II. PREREQUISITE

A. FORMULATION OF CIRCUIT DYNAMICS

Consider N_u, N_y, and N_P, the number of input signals, output signals, and circuit parameters of a nonlinear component, respectively. Also let $\hat{\boldsymbol{y}} = [\hat{y}_1 \dots \hat{y}_{N_y}]$, $\boldsymbol{u} = [u_1 \dots u_{N_u}]$ and $\boldsymbol{P} = [p_1 \dots p_{N_P}]$ be the outputs, time varying inputs and circuit parameters, respectively. In the rest of the paper, the input signals of the circuit are waveforms that have different values at each time, meaning that they are vectors of different real values. On the other hand, circuit parameters, such as capacitance, have constant values and are not changed through time. Also, the outputs of a circuit are vectors of signals that are generated based on the input signals at each time and parameters of the circuit. The characteristics of the original nonlinear circuit can be described in a nonlinear state space form as (1) where £ and ν are nonlinear functions,

$$\begin{aligned} \dot{\boldsymbol{X}}(t) &= £(\boldsymbol{X}(t), \boldsymbol{u}(t), \boldsymbol{P}, t) \\ \hat{\boldsymbol{y}}(t) &= \nu(\boldsymbol{X}(t), \boldsymbol{u}(t), t) \end{aligned} \tag{1}$$

$\boldsymbol{X} = [X_1 \dots X_{N_S}]^T$ is the vector of state variables, and N_s is the number of states [41]. In the case where the nonlinear circuit is complex (comprising numerous nonlinear components), the original nonlinear equations in (1) will be computationally complex to solve. Thus, a reduced complexity model that is easier to solve than the original complex equations is needed. It can be obtained by converting the original complex set of equations to a discrete-time set of equations with a specific sampling rate as follows [42]:

$$\hat{\boldsymbol{y}}(t^k) = \mathrm{f}(\hat{\boldsymbol{y}}(t^{k-1}) \dots \hat{\boldsymbol{y}}(t^{k-M_y}), \boldsymbol{u}(t^{k-1}) \dots \boldsymbol{u}(t^{k-M_u}), \boldsymbol{P}) \tag{2}$$

where k indicates the time index in the discrete-time domain, t^{k} is k^{th} time step, M_y and M_u are the number of delay steps of $\hat{\boldsymbol{y}}$ and $\boldsymbol{u}$, respectively, and f is a set of nonlinear functions.

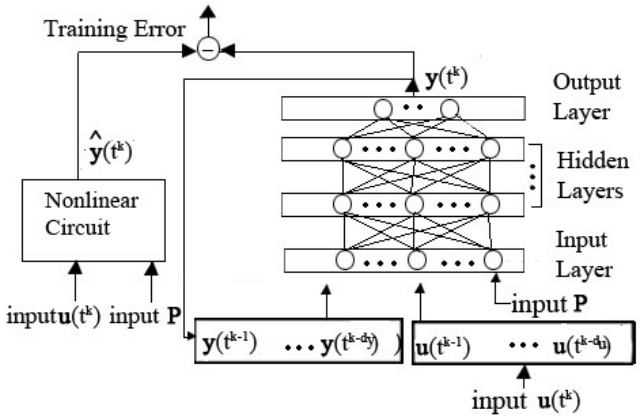

FIGURE 1. Conventional recurrent neural network structure with many hidden layers.

M_y and M_u also correspond to the number of delays for the output and input signals, respectively.

B. RECURRENT NEURAL NETWORKS (RNNs)

In this section, the RNN structure with global feedback from the output to the input (without local feedback for each hidden neuron) is presented. This has been widely used in macromodeling of nonlinear components and circuits [32], [37], [38], [39], [40], [42].

C. RNN STRUCTURE

Figure 1 demonstrates the structure of a conventional recurrent neural network. Let d_y and d_u be the number of buffers for output $\boldsymbol{y}$ and input $\boldsymbol{u}$, respectively.

In Figure 1 the first layer of the RNN includes the delayed output signals y which are returned from the output of the RNN, the delayed input signals $\boldsymbol{u}$, and the time-invariant circuit parameters $\boldsymbol{P}$. The last layer includes the time-varying output signal $\boldsymbol{y}$ which can be formulated for the i^{th} neuron at k^{th} time step as follows:

$$y_i(t^k) = \sum_{j=1}^{N_z} z_j(t^k) v_{ij} + \mu_i, \quad i = 1, \ldots .N_y \tag{3}$$

where v_{ij} is the weight between i^{th} neuron of the output layer and j^{th} neuron of the last hidden layer. The vector of the weights between the last hidden layer and the output layer, and the bias of neurons in the output layer are defined as $\boldsymbol{v} = [v_1 \ldots v_{N_y}]$ and $\boldsymbol{\mu} = [\mu_1 \ldots \mu_{N_y}]$, respectively. Also, $z_j(t^k) = \sigma(\varphi_j(t^k))$ where σ is the sigmoid activation function, z_j is the output of the j^{th} hidden neuron of the last hidden layer, and $\varphi_j(t^k)$ and N_z are the weighted summation of the outputs of the layer before the last hidden layer and the number of hidden neurons of the last hidden layer, respectively.

D. ERROR CALCULATION

The error function of the RNN, needed in the proposed method, is described here. Let $\boldsymbol{y}$(t) be the predicted output of the RNN model and $\hat{\boldsymbol{y}}$(t) be the target value. Suppose the training data be represented by input-output signals $(\boldsymbol{u}_d(t), \hat{\boldsymbol{y}}_d(t))$, $T_1 \le t \le T_2$, $d = 1 \ldots .N_{\text{w}}$, where $(\boldsymbol{u}_d(t), \hat{\boldsymbol{y}}_d(t))$ are d^{th} input-output signals, N_w is the number of input-output signals and $[T_1, T_2]$ is the input excitation interval.

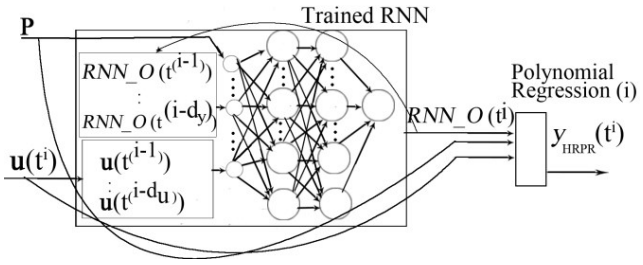

FIGURE 2. Structure of the models produced by the proposed HRPR method.

To employ a gradient-based optimization technique for training the RNN, we need the derivative of the error function with respect to the parameters of the structure. For the k^{th} time index in d^{th} training signal $\boldsymbol{u}_d((t^k), \hat{\boldsymbol{y}}_d(t^k))$, the training error is defined as:

$$E_d(t^k) = \sum_{i=1}^{N_y} \frac{1}{2} \left\{ y_{id}(t^k) - \hat{y}_{id}(t^k) \right\}^2 \tag{4}$$

where N_y is the number of output signals. Let ψ show the parameters of the RNN containing weights between layers and the bias of neurons in different layers. To compute $d_{y_{id}}(t^k)/d\psi$, the following procedure can be executed: Consider x_a as a^{th} neuron of the input layer. For $k = 1$ assume $d_{y_{id}}(t^k)/d\psi = \partial_{y_{id}}(t^k)/d\psi$, then $d_{y_{id}}(t^k)/d\psi$ for $k > 1$ can be obtained by the histories of $d_{y_{id}}(t^k)/d\psi$ as below [32], [33], [37]:

$$\frac{dy_{id}(t^k)}{d\psi} = \frac{\partial y_{id}(t^k)}{\partial \psi} + \sum_{m=1}^{L_y} \sum_{j=1}^{N_y} \left\{ \frac{\partial y_{jd}(t^k)}{\partial x_{[j+(m-1)N_y]}} \frac{dy_{jd}(t^{k-m})}{\partial \psi} \right\},$$

$$L_y = \begin{cases} d_y, \text{ if } k > d_y \\ k-1, \text{ otherwise} \end{cases} \tag{5}$$

where $x_{[j+(m-1)N_y]}$ and $y_{jd}(t^{k-m})$ are equal. The recurrent backpropagation consists of two parts. In the first part partial derivative $\partial y_{id}(t^k)/\partial \psi$ is obtained by normal back propagation through the feedforward neural network (FFNN) between the input and output layers. In the second part, $\partial y_{jd}(t^k)/\partial x_{[j+(m-1)N_y]}$ is computed by further back propagating to the input layer and can be written as:

$$\frac{\partial y_{id}(t^k)}{\partial x_{[j+(m-1)N_y]}} = \sum_{r=1}^{N_z} \frac{dy_{id}(t^k)}{dz_r} \frac{dz_r}{dx_{[j+(m-1)N_y]}}$$
$$\text{For } j = 1 \ldots .Ny; \quad m = 1 \ldots .dy \tag{6}$$

Now the derivative $\partial y_{id}(t^k)/\partial \psi$ is stored to be used as history for computing the derivative at $(k+1)^{\text{th}}$ time step.

III. THE PROPOSED HYBRID METHOD

The proposed hybrid RNN-polynomial regression (HRPR) method combines recurrent neural network and polynomial

regression models. The polynomial models use outputs of the RNN at their inputs. These two parts of the models developed with the HRPR method have separate structures and full training should be done in such a way that training of RNN should be performed before training of the polynomial models.

A. HYBRID RNN-POLYNOMIAL REGRESSION (HRPR)

In this section, the structure of the models produced by the proposed HRPR method is presented in detail. The means of combining RNN and the polynomial regression model is explained in section III.B. Figure 2 indicates the structure of the proposed hybrid method which consists of two parts: the first part is a trained conventional RNN structure that receives the input signal(s) and circuit parameters of electronic component and generates the output signals partly used as inputs of the next part. The second part consists of polynomial regressions receiving the output of the RNN, circuit parameters, and the input(s) of the electronic component simultaneously as their inputs in order to generate the final output. As shown in Figure 2, each time step in the proposed method has its own polynomial regression unit. Assume there are S time steps in each waveform. In Figure 2 $\mathbf{u}(t^i)$, $y_{HRPR}(t^i)$, $RNN\_O(t^i)$ $(i = 1 \dots S)$, and $\boldsymbol{P}$ are input and output signals of electronic component at time step i^{th}, output of RNN at time step i^{th}, and circuit parameters, respectively. In this way, each external input signal at the i^{th} time step and circuit parameters, are first passed through the RNN to generate its outputs at the same time step, and then, the generated output along with external inputs and circuit parameter are given to their corresponding polynomial regression unit (i^{th} unit). Finally, the output of the model is obtained by S polynomial regression units.

B. POLYNOMIAL REGRESSION

1) POLYNOMIAL REGRESSION IN A LINEAR MODEL

As mentioned in section III.A, the second part of the proposed HRPR method consists of polynomial regressions. Noteworthy to mention that in machine learning area, polynomial regression can be used in a linear model as discussed in [52]. Let us first explain the linear model concept as defined in [52] in the following paragraphs.

The simplest linear model for regression involves a linear combination of the input variables:

$$y(\boldsymbol{x}, \boldsymbol{w}) = w_0 + \sum_{j=1}^{D+1} w_j x_j \tag{7}$$

where D is the sum of the number of input signals and circuit parameters of the nonlinear component to be modeled and $\boldsymbol{x} = (x_1 \dots x_{D+1})$ is the vector of input variables. This is often simply known as linear regression. The main property of this model is that it is a linear function of the regression parameters, $w_0, \dots, w_{D+1}$. It is also, a linear function of the input variables x_j, and this causes the model to have significant limitations. Thus, we extend the class of linear models by considering linear combinations of fixed nonlinear functions of the input variables:

$$y(\boldsymbol{x}, \boldsymbol{w}) = w_0 + \sum_{j=1}^{\hat{D}+1} w_j \zeta_j(\boldsymbol{x}) \tag{8}$$

where ζ_j is known as basis-function and $\hat{D}$ is the number of basis-functions. Indeed, the basis-functions enable the model to entangle a function of the input instead of the input itself in order to increase the ability of the model in capturing more complex relationships between input and output. By denoting the maximum value of the index j by $\hat{D}+1$, the total number of regression parameters in this model will be $\hat{D}+2$ and the parameter W_0 corresponds to any fixed offset in the data and is called bias [52].

Now we can use any nonlinear function as basis-function in (8). As an example, for an input vector $\boldsymbol{x} = (x_1, x_2)^{\mathrm{T}}$, if we set $\zeta_1(\boldsymbol{x}) = x_1$ and $\zeta_2(\boldsymbol{x}) = x_2$, a polynomial regression with degree 1 is formed as bellow:

$$y(\boldsymbol{x}, \boldsymbol{w}) = w_0 + w_1 x_1 + w_2 x_2 \tag{9}$$

Also, to form multivariable polynomial regression of higher degrees, for example with degree 2, we can set basis-functions as below:

$$\begin{aligned} \zeta_1(\boldsymbol{x}) &= x_1 \\ \zeta_2(\boldsymbol{x}) &= x_2 \\ \zeta_3(\boldsymbol{x}) &= (x_1)^2 \\ \zeta_4(\boldsymbol{x}) &= (x_2)^2 \\ \zeta_5(\boldsymbol{x}) &= x_1 x_2 \end{aligned} \tag{10}$$

Consequently, a linear model of the polynomial regression with degree 2 based on (8) is constructed as below:

$$\begin{aligned} y(\boldsymbol{x}, \boldsymbol{w}) = w_0 + w_1 x_1 + w_2 x_2 + w_3 (x_1)^2 \\ + w_4 (x_2)^2 + w_5 x_1 x_2 \end{aligned} \tag{11}$$

We can see in (11) that a polynomial regression can be used in linear models where the output is linear in terms of $\boldsymbol{w}$[52]. In fact, we can have both linear and nonlinear polynomial regression models. In this paper, the linear polynomial regression model has been used in the proposed method.

2) POLYNOMIAL REGRESSION IN THE PROPOSED METHOD

To use polynomial regression in each time step of the HRPR method same as Figure 2, the equation (8) is rewritten as below:

$$y_{HRPR}(t^i) = y_{PR(i)}(\boldsymbol{x}^i, \boldsymbol{w}^i) = w_0^i + \sum_{j=1}^{\hat{D}+1} w_j^i \zeta_j(\boldsymbol{x}^i) \tag{12}$$

where $y_{PR(i)}$ is the output of i^{th} polynomial regression unit and $\boldsymbol{x}^i = [RNN\_O(t^i), \boldsymbol{P}, \boldsymbol{u}(t^i)]^T$ in which ζ_j should be defined accordingly to form a polynomial regression with the desired degree. Also $\boldsymbol{w}^i = [w_1 \dots w_{\hat{D}}]$ is a vector containing parameters of the polynomial regression where $\hat{D}$, which is

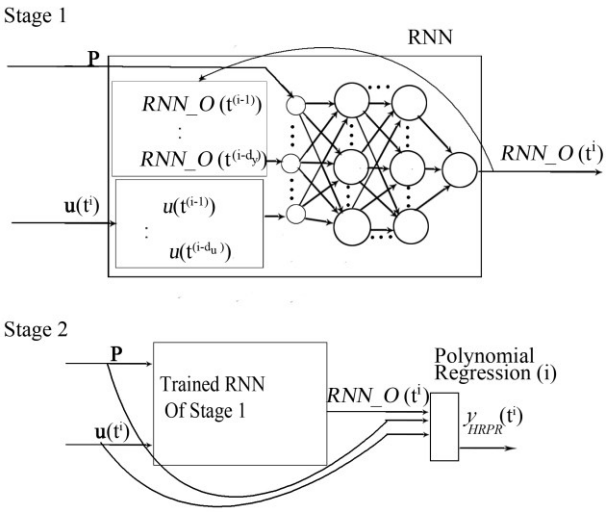

FIGURE 3. Training Stages of the Proposed HRPR Method.

the number of terms in the constructed polynomial regression, depends on the degree of the polynomial. For example, in (12), if polynomial regression with degree 1 is used, $\hat{D}$ will be 3 and if polynomial with degree 2 is used, $\hat{D}$ will be 10.

C. ORDINARY LEAST SQUARE (OLS)

In section III.B the polynomial regression which can be used in a linear model was introduced. Iterative optimization methods such as stochastic gradient descent (SGD) can be used to find the parameters of the linear model. Other than iterative methods, there are closed-form methods that do not require parameters such as learning rate and number of epochs used in the iterative form. In this section, the Ordinary Least Square (OLS) is introduced. It is one of the well-known closed-form solutions in linear model estimation used in the literature [58], [59].

Assume there exists n number of samples for solving a polynomial regression. For simplicity consider a polynomial regression with degree 1 as below:

$$y_{PR(i)}(\boldsymbol{x}^i, \boldsymbol{w}^i) = w_0^i + w_1^i x_1^i + \cdots + w_{D+1}^i x_{D+1}^i \tag{13}$$

We can put all together in the form of $\boldsymbol{Y}^i = [y_1^i, y_2^i, \ldots, y_n^i]^T$ where y_j^i $(j = 1 \ldots n)$ is target output of j^{th} sample in i^{th} time step, and coefficients at time step i^{th} can be written as $\boldsymbol{W}^i = \left[w_1^i, \ldots, w_{D+1}^i\right]^T$ (bias is ignored for simplicity). Also, let n samples at i^{th} time step be noted as:

$$\boldsymbol{X}^i = \begin{bmatrix} x_{11}^i x_{12}^i \ldots x_{1D+1}^i \\ x_{21}^i x_{22}^i \ldots x_{2D+1}^i \\ \vdots \\ x_{n1}^i x_{n2}^i \ldots x_{nD+1}^i \end{bmatrix} \tag{14}$$

where each row of $\boldsymbol{X}^i$ is devoted to one sample. Now we should find parameters of $\boldsymbol{W}^i$ such that the following objective function is minimized:

$$\hat{\boldsymbol{W}}^i = \arg\min_{\boldsymbol{W}^i} \mathrm{J}(\boldsymbol{W}^i) \tag{15}$$

where $\mathrm{J}(\mathbf{W}^i)$ is the objective function defined as:

$$\mathrm{J}(\boldsymbol{W}^i) = \sum_{k=1}^{n} \left| \sum_{j=1}^{D+1} x_{kj}^i w_j^i - y_k^i \right|^2 = \left\| \boldsymbol{X}^i \boldsymbol{W}^i - \boldsymbol{Y}^i \right\|^2 \tag{16}$$

Due to the fact that power of two of a matrix is equal to multiplication of the transpose of that with itself, $\mathrm{J}(\boldsymbol{W}^i)$ can be expanded to:

$$\begin{aligned} J(\boldsymbol{W}^i) &= \left(\boldsymbol{X}^i \boldsymbol{W}^i - \boldsymbol{Y}^i\right)^T \left(\boldsymbol{X}^i \boldsymbol{W}^i - \boldsymbol{Y}^i\right) \\ &= (\boldsymbol{Y}^i)^T \boldsymbol{Y}^i - (\boldsymbol{Y}^i)^T \boldsymbol{X}^i \boldsymbol{W}^i \\ &\quad - (\boldsymbol{W}^i)^T (\boldsymbol{X}^i)^T \boldsymbol{Y}^i + (\boldsymbol{W}^i)^T (\boldsymbol{X}^i)^T \boldsymbol{X}^i \boldsymbol{W}^i \end{aligned} \tag{17}$$

Finally, gradients can be calculated as (18), shown at the bottom of the next page, in order to find optimal $\boldsymbol{W}^i$, the gradients should be set to zero. Therefore,

$$\begin{aligned} -2(\boldsymbol{Y}^i)^T \boldsymbol{X}^i + 2(\boldsymbol{W}^i)^T (\boldsymbol{X}^i)^T \boldsymbol{X}^i &= 0 \\ \Rightarrow (\boldsymbol{Y}^i)^T \boldsymbol{X}^i &= (\boldsymbol{W}^i)^T (\boldsymbol{X}^i)^T \boldsymbol{X}^i \\ \Rightarrow (\boldsymbol{X}^i)^T \boldsymbol{Y}^i &= (\boldsymbol{X}^i)^T \boldsymbol{X}^i \boldsymbol{W}^i \\ \Rightarrow \boldsymbol{W}^i &= \left((\boldsymbol{X}^i)^T \boldsymbol{X}^i\right)^{-1} (\boldsymbol{X}^i)^T \boldsymbol{Y}^i \end{aligned} \tag{19}$$

where $\left((\boldsymbol{X}^i)^T \boldsymbol{X}^i\right)^{-1}$ demonstrates the inverse of matrix $\left((\boldsymbol{X}^i)^T \boldsymbol{X}^i\right)^{-1}$. Using (19), parameters of the linear model

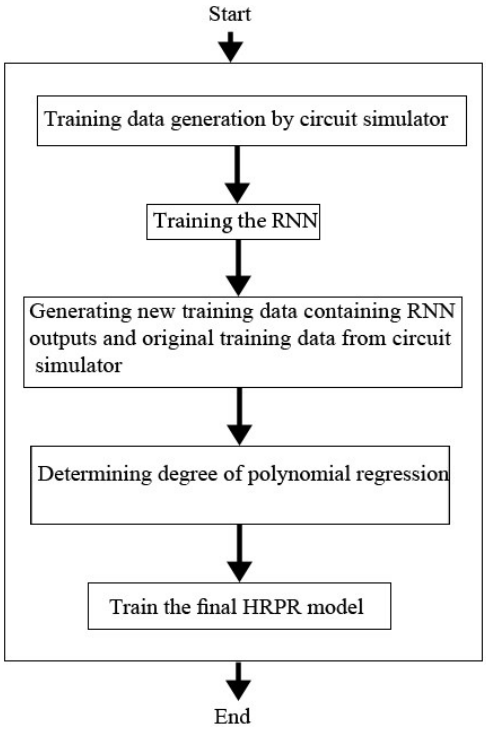

FIGURE 4. Flowchart of training procedure for the proposed HRPR method.

are found based on inputs and outputs of the training data. It means that parameters are learned directly without performing many epochs and the model is ready to be used for test data.

Therefore, because of the closed-form nature of the OLS method, it is much faster than iterative methods and setting free parameters such as iteration number and learning rate is not required.

D. DETERMINING THE POLYNOMIAL DEGREE IN HRPR

The OLS formulas introduced in section III.C are based on equation (13) which is a polynomial regression with degree 1. Polynomial regression with higher degrees can also be used in the OLS method. This is because, as discussed earlier in section III.B, polynomial regression with higher degrees can be also linear in terms of $\boldsymbol{W}^i$ coefficients. To use polynomial regression with higher degrees in OLS, matrix of n samples similar to equation (14) should be created based on polynomial regression which is represented in equation (20) (ignoring bias for simplicity) for polynomial of degree 2 and $D = 1$, and then replaced in equation (16) and corresponding coefficients will be the vector of $\boldsymbol{W}^i = \left[w_1^i, \ldots, w_{\hat{D}}^i\right]^T$, where $\hat{D}$ is the number of variables in each row of (20). The rest of the procedure is similar for polynomial regressions with degrees 1 and 2 in HRPR. For polynomial regression with degree 1, we use equations (14)-(19) and if polynomial with degree 2 is used, the same equations are needed, except equation (14) which should be replaced with (20). Each row in the matrix of equation (20) corresponds to one training data for i^{th} polynomial regression model where x_{m1}^i and x_{m2}^i are the output of RNN and the external input of circuit at i^{th} time step respectively for m^{th} training data.

$$\boldsymbol{X}^i = \begin{bmatrix} x_{11}^i\, x_{12}^i\, (x_{11}^i)^2\, (x_{12}^i)^2\, x_{11}^i x_{12}^i \\ \vdots \\ x_{n1}^i\, x_{n2}^i\, (x_{n1}^i)^2\, (x_{n2}^i)^2\, x_{n1}^i x_{n2}^i \end{bmatrix} \tag{20}$$

E. TRAINING OF THE RNN WITH THE PROPOSED HRPR METHOD

As shown in Figure 3 training of the proposed method is done in two stages: In the first stage, the training waveforms are obtained using circuit simulation tool and an RNN is trained using these waveforms.

In second stage, as shown in Figure 3, outputs obtained from trained RNN in addition to the input(s) of nonlinear component in each time step and circuit parameters, are concatenated to form the training data for the polynomial regression in the same time step. Therefore, for each time step, a specific regressor is trained according to equation (19). Suppose we have n training waveforms, each containing S time steps. After training the RNN using these training waveforms, the equation (13) can be rewritten as (21).

$$y_{HRLM(i)}(.) = w_0^i + w_1^i RNN\_O(t^i) + w_2^i u_2^i + \cdots + w_{D+1}^i u_{D+1}^i \quad 1 \le i \le S \tag{21}$$

where $RNN\_O(t^i)$ is the output of RNN at time step i^{th} based on equation (3) and $u_2^i, \ldots, u_{D+1}^i$ are circuit parameters and input signals of nonlinear component all in time step i^{th}, respectively, and $W_0^i, \ldots, u_{D+1}^i$ are parameters of polynomial regression unit at time step i^{th}. Noteworthy to mention that the RNN structure used in the proposed method has considerably fewer parameters compared to the conventional RNN structure for modeling the same component. The flowchart in Figure 4 demonstrates the training procedure of the proposed HRPR method.

IV. NUMERICAL RESULTS

A. TRANSMISSION-GATE CIRCUIT

The first example to verify the validity of the proposed method is the Transmission-Gate (TG) component shown in Figure 5. Training and test waveforms were generated using the SPICE circuit simulator. A set of signals were generated as training data by varying rise/fall times from 50ps to 60ps with steps of 2ps and load capacitance of 20-24fF with steps of 2fF.

Some other signals with rise/fall times of 51ps to 57ps with steps of 2ps and load capacitance of 20.5, 21, 23, and 23.5 fF were generated as test waveforms which were not used in the training procedure. Table 1 shows the comparison of the training and the test errors/times using the proposed HRPR, conventional RNN, and LSTM [36] methods for modeling TG circuit. The results prove that the proposed hybrid method achieves good enough accuracy in much less training time compared to the conventional RNN and LSTM methods.

Also, the test time of the proposed HRPR-based model is less than conventional RNN-based and LSTM-based models. The comparison of output test signals obtained using the proposed method, the RNN-based model, and Transistor- level models are shown in Figure 6. Also, Table 2 represents the simulation (test) time speedup of the transistor-level and the proposed HRPR-based. As it can be seen from the table, the model obtained from the proposed technique for TG component is considerably faster than the existing model in circuit simulators.

$$\frac{\partial \mathrm{J}}{\partial \boldsymbol{W}^i} = \frac{\partial \left((\boldsymbol{Y}^i)^T \boldsymbol{Y}^i - (\boldsymbol{Y}^i)^T \boldsymbol{X}^i \boldsymbol{W}^i - (\boldsymbol{W}^i)^T (\boldsymbol{X}^i)^T \boldsymbol{Y}^i + (\boldsymbol{W}^i)^T (\boldsymbol{X}^i)^T \boldsymbol{X}^i \boldsymbol{W}^i\right)}{\partial \boldsymbol{W}^i} = -2(\boldsymbol{Y}^i)^T \boldsymbol{X}^i + 2(\boldsymbol{W}^i)^T (\boldsymbol{X}^i)^T \boldsymbol{X}^i \tag{18}$$

TABLE 1. Comparison between a conventional RNN, LSTM, and the RNN at the core of models derived with the proposed HRPR method when modeling transmission-gate.

Model type	Structure	Number of parameters	Training error	Testing error	Training time	Test time speedup
Conventional RNN-based	Hidden Neurons=15, Layer=2, Delay=4	416	1.86×10^{-5}	14.4×10^{-5}	3655s	1(reference)
RNN at the core of models derived with the HRPR-method	Hidden Neurons=20, Layer=1, Delay=2, Degree of Polynomial Regression=1	145	6.69×10^{-5}	7.22×10^{-5}	50s	2.4
LSTM	Cells=6, Layer=2	535	2.5×10^{-5}	7.3×10^{-5}	3550s	0.77

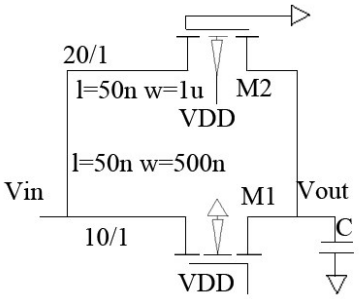

FIGURE 5. Schematic of TG circuit.

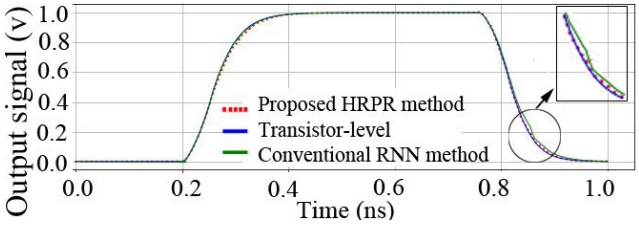

FIGURE 6. Comparison of the outputs (test data) generated by proposed HRPR-based, RNN-based, and transistor-level models for transmission-gate component.

TABLE 2. Comparison of CPU time speedup for the HRPR-based and transistor-level models of the transmission gate device.

Model type	CPU Time (ms)
HRPR-based	39
Transistor-Level	280

B. FREQUENCY DOUBLER DEVICE

The schematic of a frequency doubler has been shown in Figure 7. A set of signals were generated as training waveforms by varying frequency from 2 kHz to 2.1 kHz with steps of 0.02 kHz and amplitudes of 0.09, 0.1 and 0.11 Volts. Some other signals with frequencies of 2.01, 2.03, and 2.05 kHz and amplitudes of 0.092, 0.094, 0.098, 0.106, and 0.108 Volts were generated as test data. Table 3 shows the comparison of training and test errors/times using the proposed HRPR, the conventional RNN, and LSTM methods for modeling frequency doubler component. As it can be seen in this table, polynomial regression with degree 1 is not capable of achieving the desired accuracy in training/test procedures but polynomial regression with degree 2, due to having more

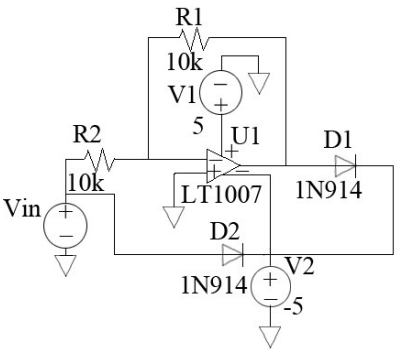

FIGURE 7. Schematic of the frequency doubler used in this paper.

complex structure, can perform better on both training and test processes.

The comparison of output test signals obtained using the proposed HRPR-based, RNN-based, and transistor-level models are shown in Figure 8. Table 4 shows the speedup comparison among the transistor-level, the proposed HRPR-based, and LSTM-based models for frequency doubler.

As can be seen from Tables 3 and 4 the proposed hybrid modeling method demonstrates significantly less training time in comparison with the conventional RNN and LSTM methods. Also, the final obtained HRPR-based model of frequency doubler not only shows considerable speedup compared to the transistor-level model but is also faster to compute than the model obtained using conventional RNN and LSTM technique.

C. CMOS INVERTER

The schematic of a CMOS inverter used in this paper is shown in Figure 9. A set of signals was generated as training data by varying the rise/fall times from 1.6ps to 2.6ps with steps of 0.2ps and amplitudes of 0.9V, 1V, and 1.1V. Some other signals with rise/fall times of 1.9ps to 2.5ps with step0.2ps and amplitudes of 0.92, 0.94, 0.95, 0.96, 0.98, 0.102, and 0.108 Volt were generated as test data.

Also, comparison of training and test errors and times using HRPR, conventional RNN, and LSTM methods for modeling a CMOS inverter is shown in Table 5. As can be seen from the table, the proposed HRPR method is remarkably faster to train compared to conventional RNN and LSTM models. Also, the final model of the CMOS inverter obtained from

TABLE 3. Comparison between conventional RNN, LSTM, and the proposed HRPR method for modeling frequency doubler.

Model type	Structure	Number of parameters	Train error	Test error	Train time	Test time speedup
RNN-based	Hidden Neurons=15, Layer=2, Delay=5	421	5.26×10^{-5}	9.54×10^{-5}	7148s	1(reference)
RNN at the core of models derived with the HRPR-method	Hidden Neurons=20, Layer=1, Delay=2, Degree of Polynomial Regression=1	124	17×10^{-5}	19.1×10^{-5}	1069.5s	3.11
RNN at the core of models derived with the HRPR-method	Hidden Neurons=20, Layer=1, Delay=2, Degree of Polynomial Regression=2	127	7.32×10^{-5}	7.92×10^{-5}	1070s	3.1
LSTM	Cells=7, Layer=2	680	6.2×10^{-5}	8.1×10^{-5}	5410s	0.56

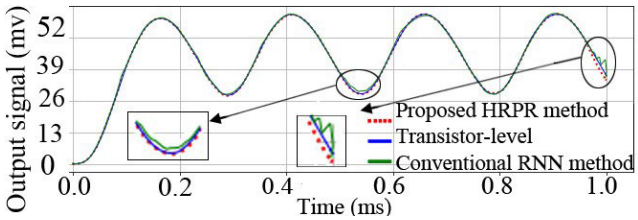

FIGURE 8. Comparison of the outputs (test data) generated by proposed HRPR-based, RNN-based, and Transistor-level models for frequency doubler component.

TABLE 4. CPU time speedup comparison between the HRPR-based, and transistor-level models of the frequency doubler.

Model type	CPU Time (ms)
HRPR-based	41
Transistor-Level	418

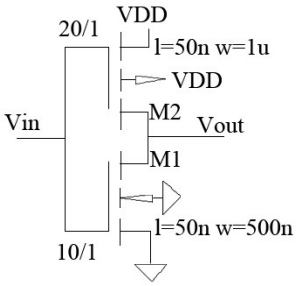

FIGURE 9. Schematic of CMOS inverter device.

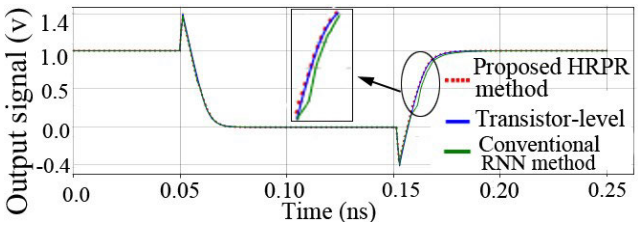

FIGURE 10. Comparison of the outputs (test data) generated by the proposed HRPR-based, RNN-based, and Transistor-level models for a CMOS inverter.

the HRPR technique is much faster to compute than the one obtained using the RNN and LSTM techniques. The output test signals of the proposed HRPR-based, RNN-based, and Transistor-level models are shown in Figure 10. These results show that the proposed HRPR-based model better matches the transistor-level model than the RNN-based model. Table 6 also demonstrated the speedup achieved using the transistor-level and the proposed HRPR- based models for the CMOS inverter component. The results in this table show considerable CPU time improvement for the HRPR-based model in comparison with the transistor-level model.

As it was seen in all three examples, models produced by the HRPR method outperform the conventional straight RNN models in terms of both accuracy and speed. Indeed, according to equation (21), regression units of stage 2 fine-tune the incoming outputs from RNN of stage 1, resulting in better accuracy. The presented results demonstrate that the proposed method needs considerably less training time compared to the conventional RNN and LSTM methods in order to model the same circuits without losing accuracy. Also, as RNN of stage 1 contains considerably a smaller number of parameters compared to the conventional RNN and LSTM, a great speedup has been achieved. It means that using the proposed method for modeling nonlinear circuits leads to a faster model which generates the output in a shorter time.

Noteworthy to mention that the proposed method is not limited to transient simulation of components and circuits. It can be applied for any time-domain (transient or steady state) simulations. The limitation here is the availability of accurate training data, so if we want to use this technique for periodic steady state analysis, the proposed method has to be trained for that which means we should generate enough steady state training data and pass them to the proposed method to be trained. Since the neural network-based models, such as the model obtained by the proposed HRPR method, are data-dependent models, the functionality of the model directly depends on the generated training data. For example, if we train the model in a specified range of data, we only expect the model to work well on testing data inside the range and for having a suitable model outside of the range,

TABLE 5. Comparison between a conventional RNN, LSTM, and the proposed HRPR method for modeling a CMOS inverter.

Model type	Structure	Number of parameters	Training error	Testing error	Training time	Testing time speedup
Conventional RNN-based	Hidden Neurons=30, Layer=2, Delay=6	1351	4.3×10^{-5}	14×10^{-5}	18535s	1(reference)
RNN at the core of models derived with the HRPR-method	Hidden Neurons=25, Layer=1, Delay=2, Degree of Polynomial Regression=1	154	8.9×10^{-5}	11.8×10^{-5}	205s	6.1
LSTM	Cells=11, Layer=2	1596	4.9×10^{-5}	11.2×10^{-5}	10200s	0.82

TABLE 6. CPU time speedup comparison between a CMOS inverter model produced by the HRPR method and a transistor-level model of a CMOS inverter.

Model type	CPU Time (ms)
HRPR-based	44
Transistor-Level	380

TABLE 7. Comparison between the number of training waveforms required for the HRPR and RNN methods when creating models with similar accuracy.

Method	Number of training waveforms	Train error	Test error
RNN	18	1.86×10^{-5}	14.4×10^{-5}
HRPR	9	4.89×10^{-5}	12.8×10^{-5}

TABLE 8. The effect of noisy data on modeling TG circuit using proposed HRPR method.

	Clean train and clean test	Noisy train and noisy test	Noisy train and clean test
Training error	6.69×10^{-5}	8.8×10^{-5}	8.8×10^{-5}
Testing error	7.22×10^{-5}	11.5×10^{-5}	10×10^{-5}

we should expand the initial range for generating the training data.

D. DATA REDUCTION IN MODEL DEVELOPMENT

One of the main concerns in modeling nonlinear components is the amount of data required for developing an accurate model as generating data is usually costly. The proposed HRPR technique not only creates more accurate models but also requires a smaller number of training data for creating models with similar accuracy compared to the conventional RNN method. To show efficiency of the proposed HRPR method in this case, Transmission Gate example was trained again with different number of training waveforms and the results have been shown in Table 7. As it can be seen from Table 7, the proposed HRPR method requires half of the number of training waveforms compared to the conventional RNN for developing model with similar accuracy for this device.

E. ROBUSTNESS OF THE PRPOSED METHOD AGAINTS NOISY DATA

Training the model based on the HRPR method can be performed by the training data which have been generated by simulation software or measurement tools. In the case of generating training data by measurement tools, we will likely encounter noise in the data. So, a set of noisy data have been generated to train the model based on HRPR method. The noisy data are generated by applying an additional white noise (Gaussian noise) to the original waveforms [36], [47]. Table 8 demonstrates the effect of noise on the functionality of the proposed HRPR method for modeling the TG circuit. As can be seen in this table, despite slightly degenerating the accuracy of the HRPR-based model by the appearance of the noise in the data, an acceptable accuracy is still obtained. These results demonstrate the robustness of the proposed HRPR-based model against the noisy data.

V. CONCLUSION

In this paper a hybrid method called HRPR was proposed. The method combines conventional RNN and polynomial regression models. HRPR was used for modeling nonlinear electronic components. Input-output waveforms of the original component that were used in the training and testing procedures were obtained from SPICE circuit simulator. The new method demonstrated significant reduction in training time compared to models obtained from the conventional RNN and LSTM modeling method due to its more efficient structure and training procedure. Also, it showed considerable speedup in inference (simulation) time compared to the models obtained from the conventional RNN and LSTM methods and the transistor-level models in existing circuit simulation tools. Additionally, the models obtained from the proposed structure introduce less inference time errors compared to the models obtained from the conventional RNN structure. Using the proposed HRPR method for modeling nonlinear components, there is no need to have deep knowledge of details of the internal structure of each component or device. In addition to these advantages, the proposed hybrid method required fewer training waveforms compared to the conventional RNN method to create models with similar accuracy. Three practical examples were used in this paper to demonstrate the validity of the proposed macromodeling approach.

REFERENCES

[1] N. Mirzaie and R. Rohrer, "A macromodeling approach for analog behavior of digital integrated circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 12, pp. 5025–5031, Dec. 2020.
[2] W. Liu, W. Na, L. Zhu, J. Ma, and Q.-J. Zhang, "A wiener-type dynamic neural network approach to the modeling of nonlinear microwave devices," *IEEE Trans. Microw. Theory Techn.*, vol. 65, no. 6, pp. 2043–2062, Feb. 2017.

[3] F. Feng, W. Na, J. Jin, J. Zhang, W. Zhang, and Q.-J. Zhang, "Artificial neural networks for microwave computer-aided design: The state of the art," *IEEE Trans. Microw. Theory Techn.*, vol. 70, no. 11, pp. 4597–4619, Nov. 2022.

[4] W. Liu, F. Feng, Y. Zhuo, J. Zhang, Q. Lin, K. Ma, and Q.-J. Zhang, "Electromagnetic parametric modeling using combined neural networks and RLGC-based eigenfunctions for two-port microstrip structures," *IEEE Trans. Microw. Theory Techn.*, vol. 70, no. 11, pp. 4666–4682, Nov. 2022.

[5] F. Feng, W. Na, W. Liu, S. Yan, L. Zhu, J. Ma, and Q.-J. Zhang, "Multifeature-assisted neuro-transfer function surrogate-based EM optimization exploiting trust-region algorithms for microwave filter design," *IEEE Trans. Microw. Theory Techn.*, vol. 68, no. 2, pp. 531–542, Feb. 2020.

[6] J. Zhang, S. Yan, F. Feng, J. Jin, W. Zhang, J. Wang, and Q.-J. Zhang, "A novel surrogate-based approach to yield estimation and optimization of microwave structures using combined quadratic mappings and matrix transfer functions," *IEEE Trans. Microw. Theory Techn.*, vol. 70, no. 8, pp. 3802–3816, Aug. 2022.

[7] Y. Zhuo, F. Feng, J. Zhang, and Q.-J. Zhang, "Parametric modeling incorporating joint polynomial-transfer function with neural networks for microwave filters," *IEEE Trans. Microw. Theory Techn.*, vol. 70, no. 11, pp. 4652–4665, Nov. 2022.

[8] F. Zouari, K. Ben Saad, and M. Benrejeb, "Adaptive backstepping control for a class of uncertain single input single output nonlinear systems," in *Proc. 10th Int. Multi-Conf. Syst., Signals Devices (SSD)*, Mar. 2013, pp. 1–6.

[9] F. Zouari, K. B. Saad, and M. Benrejeb, "Robust adaptive control for a class of nonlinear systems using the backstepping method," *Int. J. Adv. Robotic Syst.*, vol. 10, no. 3, p. 166, Mar. 2013.

[10] S. Wang, J. Na, and X. Ren, "Rise-based asymptotic prescribed performance tracking control of nonlinear servo mechanisms," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 12, pp. 2359–2370, Dec. 2018.

[11] J. Na, Y. Huang, X. Wu, S. Su, and G. Li, "Adaptive finite-time fuzzy control of nonlinear active suspension systems with input delay," *IEEE Trans. Cybern.*, vol. 50, no. 6, pp. 2639–2650, Jun. 2020.

[12] S. Wang, X. Ren, J. Na, and T. Zeng, "Extended-state-observer-based funnel control for nonlinear servomechanisms with prescribed tracking performance," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 1, pp. 98–108, Jan. 2017.

[13] D. De Jonghe and G. Gielen, "Efficient analytical macromodeling of large analog circuits by transfer function trajectories," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, San Jose, CA, USA, Nov. 2011, pp. 91–94.

[14] D. De Jonghe, D. Deschrijver, T. Dhaene, and G. Gielen, "Extracting analytical nonlinear models from analog circuits by recursive vector fitting of transfer function trajectories," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, 2013, pp. 1448–1453.

[15] L. Zhu, Q. Zhang, K. Liu, Y. Ma, B. Peng, and S. Yan, "A novel dynamic neuro-space mapping approach for nonlinear microwave device modeling," *IEEE Microw. Wireless Compon. Lett.*, vol. 26, no. 2, pp. 131–133, Feb. 2016.

[16] D. Marche and Y. Savaria, "Modeling R-2R segmented-ladder DACs," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 57, no. 1, pp. 31–43, Jan. 2010.

[17] S. R. Hasan, N. Bélanger, Y. Savaria, and M. O. Ahmad, "Crosstalk glitch propagation modeling for asynchronous interfaces in globally asynchronous locally synchronous systems," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 57, no. 8, pp. 2020–2031, Aug. 2010.

[18] W. Dghais and J. Rodriguez, "Empirical modelling of FDSOI CMOS inverter for signal/power integrity simulation," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, 2015, pp. 1555–1558.

[19] O. Rahmani, S. A. Sadrossadat, S. M. Mirimani, and S. H. Mirimani, "Optimization of efficiency and output power of 8/6 switched reluctance motor using new neural network-based adjoint Lp metric method," *IET Electric Power Appl.*, vol. 15, no. 6, pp. 769–783, Jun. 2021.

[20] S. A. Sadrossadat and O. Rahmani, "A framework for statistical design of a brushless DC motor considering efficiency maximization," *IET Electr. Power Appl.*, vol. 16, no. 3, pp. 407–420, Mar. 2022.

[21] J. Jin, F. Feng, J. Zhang, S. Yan, W. Na, and Q. Zhang, "A novel deep neural network topology for parametric modeling of passive microwave components," *IEEE Access*, vol. 8, pp. 82273–82285, 2020.

[22] Z. Mirzadeh, J.-F. Boland, and Y. Savaria, "Modeling the faulty behaviour of digital designs using a feed forward neural network approach," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Lisbon, Purtugal, May 2015, pp. 1518–1521.

[23] M. AskariHemmat, O. Bilaniuk, S. Wagner, Y. Savaria, and J.-P. David, "RISC-V barrel processor for deep neural network acceleration," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Daegu, South Korea, May 2021, pp. 1–5.

[24] A. Ghaffari and Y. Savaria, "CNN2Gate: An implementation of convolutional neural networks inference on FPGAs with automated design space exploration," *Electronics*, vol. 9, no. 12, p. 2200, Dec. 2020.

[25] S. Wang, H. Yu, J. Yu, J. Na, and X. Ren, "Neural-network-based adaptive funnel control for servo mechanisms with unknown dead-zone," *IEEE Trans. Cybern.*, vol. 50, no. 4, pp. 1383–1394, Apr. 2020.

[26] J. Na, S. Wang, Y.-J. Liu, Y. Huang, and X. Ren, "Finite-time convergence adaptive neural network control for nonlinear servo systems," *IEEE Trans. Cybern.*, vol. 50, no. 6, pp. 2568–2579, Jun. 2020.

[27] F. Zouari, "Adaptive internal model control of a DC motor drive system using dynamic neural network," *J. Softw. Eng. Appl.*, vol. 5, no. 3, pp. 168–189, 2012.

[28] A. Rawat, R. N. Yadav, and S. C. Shrivastava, "Neural network applications in smart antenna arrays: A review," *AEU-Int. J. Electron. Commun.*, vol. 66, no. 11, pp. 903–912, 2012.

[29] S. A. Sadrossadat, P. Gunupudi, and Q.-J. Zhang, "Nonlinear electronic/photonic component modeling using adjoint state-space dynamic neural network technique," *IEEE Trans. Compon., Packag., Manuf. Technol.*, vol. 5, no. 11, pp. 1679–1693, Nov. 2015.

[30] S. A. Sadrossadat, Y. Cao, and Q.-J. Zhang, "Parametric modeling of microwave passive components using sensitivity-analysis-based adjoint neural-network technique," *IEEE Trans. Microw. Theory Techn.*, vol. 61, no. 5, pp. 1733–1747, May 2013.

[31] O. Okobiah, S. P. Mohanty, and E. Kougianos, "Kriging bootstrapped neural network training for fast and accurate process variation analysis," in *Proc. 15th Int. Symp. Quality Electron. Design*, Santa Clara, CA, USA, Mar. 2014, pp. 365–372.

[32] Y. Cao and Q.-J. Zhang, "A new training approach for robust recurrent neural-network modeling of nonlinear circuits," *IEEE Trans. Microw. Theory Techn.*, vol. 57, no. 6, pp. 1539–1553, Jun. 2009.

[33] J. Xu, M. C. E. Yagoub, R. Ding, and Q.-J. Zhang, "Neural-based dynamic modeling of nonlinear microwave circuits," *IEEE Trans. Microw. Theory Techn.*, vol. 50, no. 12, pp. 2769–2780, Dec. 2002.

[34] T. Liu, S. Boumaiza, and F. M. 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.

[35] M. Magerl, V. Ceperic, and A. Baric, "Echo state networks for black-box modeling of integrated circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 35, no. 8, pp. 1309–1317, Aug. 2016.

[36] M. Moradi A., S. A. Sadrossadat, and V. Derhami, "Long short-term memory neural networks for modeling nonlinear electronic components," *IEEE Trans. Compon., Packag., Manuf. Technol.*, vol. 11, no. 5, pp. 840–847, May 2021.

[37] S. Yan, C. Zhang, and Q.-J. Zhang, "Recurrent neural network technique for behavioral modeling of power amplifier with memory effects," *Int. J. RF Microw. Comput.-Aided Eng.*, vol. 25, no. 4, pp. 289–298, May 2014.

[38] Y. Fang, M. C. E. Yagoub, F. Wang, and Q.-J. Zhang, "A new macromodeling approach for nonlinear microwave circuits based on recurrent neural networks," *IEEE Trans. Microw. Theory Techn.*, vol. 48, no. 12, pp. 2335–2344, Dec. 2000.

[39] C. Zhang, S. Yan, Q. J. Zhang, and J. G. Ma, "Behavioral modeling of power amplifier with long term memory effects using recurrent neural networks," in *Proc. IEEE Int. Wireless Symp. (IWS)*, Beijing, China, Apr. 2013, pp. 1–4.

[40] K. K. Sarma and A. Mitra, "Modeling MIMO channels using a class of complex recurrent neural network architectures," *AEU Int. J. Electron. Commun.*, vol. 66, no. 4, pp. 322–331, 2012.

[41] Z. Naghibi, S. A. Sadrossadat, and S. Safari, "Dynamic behavioral modeling of nonlinear circuits using a novel recurrent neural network technique," *Int. J. Circuit Theory Appl.*, vol. 47, no. 7, pp. 1071–1085, Jul. 2019.

[42] Z. Naghibi, S. A. Sadrossadat, and S. Safari, "Time-domain modeling of nonlinear circuits using deep recurrent neural network technique," *AEU-Int. J. Electron. Commun.*, vol. 100, pp. 66–74, Feb. 2019.

[43] S. A. Sadrossadat and Z. Naghibi, "Parallelizing time-delay recurrent neural network modeling technique on multi-core architectures," in *Proc. 16th Int. Microsystems, Packag., Assem. Circuits Technol. Conf. (IMPACT)*, Taipei, Taiwan, Dec. 2021, pp. 93–96.

[44] M. Noohi, A. Mirvakili, and S. A. Sadrossadat, "Modeling and implementation of nonlinear boost converter using local feedback deep recurrent neural network for voltage balancing in energy harvesting applications," *Int. J. Circuit Theory Appl.*, vol. 49, no. 12, pp. 4231–4247, Sep. 2021.
[45] Z. Naghibi, S. A. Sadrossadat, and S. Safari, "Adjoint recurrent neural network technique for nonlinear electronic component modeling," *Int. J. Circuit Theory Appl.*, vol. 50, no. 4, pp. 1119–1129, Apr. 2022.
[46] S. A. Sadrossadat and Z. Naghibi, "Integrating parallel computation technique to state-space dynamic neural network modeling of optical fiber," in *Proc. 16th Int. Microsystems, Packag., Assem. Circuits Technol. Conf. (IMPACT)*, Taipei, Taiwan, Dec. 2021, pp. 90–92.
[47] A. Faraji, M. Noohi, S. A. Sadrossadat, A. Mirvakili, W. Na, and F. Feng, "Batch-normalized deep recurrent neural network for high-speed nonlinear circuit macromodeling," *IEEE Trans. Microw. Theory Techn.*, vol. 70, no. 11, pp. 4857–4868, Nov. 2022.
[48] F. Zouari, A. Ibeas, A. Boulkroune, J. Cao, and M. M. Arefi, "Neural network controller design for fractional-order systems with input nonlinearities and asymmetric time-varying pseudo-state constraints," *Chaos, Solitons Fractals*, vol. 144, Mar. 2021, Art. no. 110742.
[49] C. J. Zuñiga Aguilar, J. F. Gómez-Aguilar, V. M. Alvarado-Martínez, and H. M. Romero-Ugalde, "Fractional order neural networks for system identification," *Chaos, Solitons Fractals*, vol. 130, Jan. 2020, Art. no. 109444.
[50] C. J. Zúñiga-Aguilar, H. M. Romero-Ugalde, J. F. Gómez-Aguilar, R. F. Escobar-Jiménez, and M. Valtierra-Rodríguez, "Solving fractional differential equations of variable-order involving operators with mittag-leffler kernel using artificial neural networks," *Chaos, Solitons Fractals*, vol. 103, pp. 382–403, Oct. 2017.
[51] H. M. Romero Ugalde, J.-C. Carmona, J. Reyes-Reyes, V. M. Alvarado, and J. Mantilla, "Computational cost improvement of neural network models in black box nonlinear system identification," *Neurocomputing*, vol. 166, pp. 96–108, Oct. 2015.
[52] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, Aug. 2006.
[53] I. Naseem, R. Togneri, and M. Bennamoun, "Linear regression for face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 11, pp. 2106–2112, Nov. 2010.
[54] D. Bissing, M. T. Klein, R. A. Chinnathambi, D. F. Selvaraj, and P. Ranganathan, "A hybrid regression model for day-ahead energy price forecasting," *IEEE Access*, vol. 7, pp. 36833–36842, 2019.
[55] L. H. Smith, T. A. Kuiken, and L. J. Hargrove, "Evaluation of linear regression simultaneous myoelectric control using intramuscular EMG," *IEEE Trans. Biomed. Eng.*, vol. 63, no. 4, pp. 737–746, Apr. 2016.
[56] J. Wei, T. Chen, G. Liu, and J. Yang, "Higher-order multivariable polynomial regression to estimate human affective states," *Sci. Rep.*, vol. 6, no. 1, pp. 1–13, Mar. 2016.
[57] *Spice V.17.0.35*, Analog Devices, Norwood, MA, USA. [Online]. Available: https://www.analog.com/en/design-center/design-tools-and-calculators/ltspice-simulator.html
[58] C. J. Ramos, A. P. Martins, and A. S. Carvalho, "Frequency and phase-angle estimation using ordinary least squares," *IEEE Trans. Ind. Electron.*, vol. 62, no. 9, pp. 5677–5688, Sep. 2015.
[59] A. Aloisio, R. Alaggio, and M. Fragiacomo, "Dynamic identification of a masonry façade from seismic response data based on an elementary ordinary least squares approach," *Eng. Struct.*, vol. 197, Oct. 2019, Art. no. 109415.

AMIN FARAJI received the B.S. degree in computer engineering from the University of Mohaghegh Ardabili, Ardabil, Iran, in 2019, and the master's degree in artificial intelligence from the Computer Engineering Department, Yazd University, Yazd, Iran, in 2022. His research interests include computer-aided design, deep learning, and neural networks and their applications in modeling and simulation of circuits and components.

SAYED ALIREZA SADROSSADAT (Member, IEEE) received the B.Sc. degree from the University of Tehran, Tehran, Iran, in 2007, the master's degree from the University of Waterloo, Waterloo, ON, Canada, in 2010, and the Ph.D. degree from Carleton University, Ottawa, ON, Canada, in 2015. He is currently the Chair of the Artificial Intelligence Group, Computer Engineering Department, Yazd University, Yazd, Iran. His current research interests include optimization and neural network-based modeling of linear/nonlinear components and circuits, computer-aided design, deep learning, very large-scale integration design, probabilistic design, and yield maximization. He has been a Technical Reviewer for several IEEE/IET journals, such as IEEE TRANSACTIONS ON NEURAL NETWORK AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON MICROWAVE THEORY AND TECHNIQUE, IEEE TRANSACTIONS ON VARY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, *IET Microwave and Propagation*, *IET Electronic Letters*, and IEEE CANADIAN JOURNAL OF ELECTRICAL AND COMPUTER ENGINEERING.

MAHDI YAZDIAN-DEHKORDI received the B.Sc. degree in software engineering from Yazd University, Yazd, Iran, in 2006, and the M.Sc. and Ph.D. degrees in artificial intelligence from Shiraz University, Shiraz, Iran, in 2009 and 2015, respectively. He is currently an Assistant Professor with the Department of Computer Engineering, Yazd University. His major research interests include image analysis, machine vision, machine learning, and deep learning.

MORTEZA NABAVI (Member, IEEE) received the B.S. degree in computer engineering from the Amirkabir University of Technology, Tehran, Iran, in 2008, the M.A.Sc. degree in computer engineering from Boston University, Boston, MA, USA, the M.A.Sc. degree in electronics from Carleton University, Ottawa, ON, Canada, in 2012, and the Ph.D. degree from the University of Waterloo, Waterloo, ON, Canada. He joined the Research and Development Group, Sidense Company, Ottawa, where he involved in designing subthreshold one-time programmable (OTP) memories. His current research interests include subthreshold circuits and memories.

YVON SAVARIA (Fellow, IEEE) received the B.Ing. and M.Sc.A. degrees in electrical engineering from the École Polytechnique de Montréal, Canada, in 1980 and 1982, respectively, and the Ph.D. degree in electrical engineering from McGill University, Canada, in 1985. Since 1985, he has been with the École Polytechnique de Montréal, where he is currently a Professor and the Chairperson of the Department of Electrical Engineering. He has carried out work in several areas related to microelectronic circuits and microsystems, such as testing, verification, validation, clocking methods, defect and fault tolerance, effects of radiation on electronics, high-speed interconnects and circuit design techniques, CAD methods, reconfigurable computing and applications of microelectronics to telecommunications, aerospace, image processing, video processing, radar signal processing, and digital signal processing acceleration. He is currently involved in several projects that relate to aircraft embedded systems, green IT, wireless sensor network, virtual network, computational efficiency, and application specific architecture design. He holds 16 patents, has published 97 journal articles and 354 conference papers, and was the thesis advisor of 140 graduate students who completed their studies. He was the Program Co-Chairperson of ASAP'2006 and the General Co-Chair of ASAP'2007. He has been a Consultant or was sponsored for carrying research.