Hindawi Publishing Corporation
Advances in Artificial Intelligence
Volume 2015, Article ID 184318, 10 pages
http://dx.doi.org/10.1155/2015/184318

Research Article

Wavelet Network: Online Sequential Extreme Learning Machine for Nonlinear Dynamic Systems Identification

Dhiadeen Mohammed Salih,[1] Samsul Bahari Mohd Noor,[2] Mohammad Hamiruce Merhaban,[2] and Raja Mohd Kamil[2]

[1]*Department of Computer Science, Kirkuk University, Kirkuk, Iraq*
[2]*Department of Electrical and Electronic Engineering, Universiti Putra Malaysia (UPM), Serdang, Selangor 43300, Malaysia*

Correspondence should be addressed to Dhiadeen Mohammed Salih; dhiadeen@gmail.com

Received 6 May 2015; Revised 29 July 2015; Accepted 31 August 2015

Academic Editor: Jun He

A single hidden layer feedforward neural network (SLFN) with online sequential extreme learning machine (OSELM) algorithm has been introduced and applied in many regression problems successfully. However, using SLFN with OSELM as black-box for nonlinear system identification may lead to building models for the identified plant with inconsistency responses from control perspective. The reason can refer to the random initialization procedure of the SLFN hidden node parameters with OSELM algorithm. In this paper, a single hidden layer feedforward wavelet network (WN) is introduced with OSELM for nonlinear system identification aimed at getting better generalization performances by reducing the effect of a random initialization procedure.

1. Introduction

In literature, Huang et al. presented an extreme learning machine (ELM) algorithm with a single hidden layer feedforward neural network (SLFN) in [1] and has taken many researchers' interest because it offers significant advantages such as providing better generalization performance, fast learning speed, ease of implementation, and minimal human intervention [2]. The main principle of using ELM algorithm is the random initialization procedure of the SLFN input weights, biases, and activation function parameters which are either additive or radial basis functions (RBF) and the analytical determining of the output weights in a single step using least square solution.

However, the ELM algorithm is based on fixed network structure, which means if the hidden nodes activation function (e.g., RBF's centers and impact factors) parameters are initialized, then they will not be tuned during the learning phase. Therefore, random initialization of SLFN hidden node parameters may have effect on the modeling performances [3], and to improve the SLFN it requires high complexity performance and this may lead to ill condition, which means that an ELM may not be robust enough to capture variations in data [4].

Wavelet network (WN) has been applied successfully in many applications related to system identification and black-box modelling for nonlinear dynamic systems with different batch training algorithms [5–7]. The main advantages of WN are the capability of analytical initialization procedure for the hidden nodes (wavelet activation function) parameters [8]. Moreover, wavelets decomposition properties for localization in both time and frequency domains allow better generalization in nonlinear system identification problems [7].

The similarity between the wavelet decomposing theory and single hidden layer feedforward neural networks (SLFNs) inspired Cao et al. in [9, 10] to propose a structure of the composite function wavelet neural networks (CFWNN) to be used with ELM for different applications. The initialization of the wavelet function parameters, namely, translation and dilation, is done using the input data that takes into account

the domain of input space. The results on several benchmark real-world data sets showed that the proposed CFWNN method can achieve better performances.

Latterly, a new WN structure of dual wavelet activation functions in the hidden layer nodes has been introduced by Javed et al. in [3] with ELM algorithm and called summation wavelet extreme learning machine (SW-ELM). The proposed SW-ELM showed good accuracy and generalization performances by reducing the impact of a random initialization procedure where wavelets and other parameters of hidden nodes are adjusted a priori to learning.

For many industrial applications where online sequential learning algorithms are preferred over batch learning algorithms, an online sequential extreme learning machine (OSELM) algorithm has been introduced for SLFN (NN-OSELM) with additive and RBF hidden nodes functions and showed better generalization performance and fast training capability over the other well-known sequential learning algorithms [11, 12]. However, the NN-OSELM has not been yet verified in nonlinear systems identification problems for control applications. The authors in [13] stated that if the RBF network is trained from random initial weights for each subset, it could converge to a different minimum that corresponds to weights w_i different from the one corresponding to w_0. This random initialization procedure may cause unacceptable learning behaviour in system identification applications and may lead to a different open loop response of the identified system regardless of the modeling accuracy.

To overcome these differences, the RBF can be replaced by wavelet function based on a fact that wavelets are capable of controlling the order of approximation and the regularity by some of their key mathematical properties and explicit analytical initialization form [13, 14]. In this regard, the authors in [15] introduced a feedforward WN with OSELM (WN-OSELM) algorithm for nonlinear system identification where the wavelet activation function parameters initialization played a big role to ensure fast and better learning performance over NN-OSELM. However, the proposed WN-OSELM method was based on fixed numbers of the input features and the hidden nodes which is not optimal in any case.

In this paper, feedforward WN framework is introduced with OSELM (WN-OSELM) to limit the impact of random initialization of the SLFN hidden nodes parameters by using density function with recursive algorithm [16] and the input weights and biases using Nguyen Widrow approach [17]. Moreover, the optimal input features and hidden nodes are selected using sequential forward search approach (SQFS) [18] and final prediction criterion (FPEC) [19], respectively.

The simulations will be carried out on three nonlinear systems and it will take in account the optimum number of hidden nodes and the input features for both WN and NN to ensure the generalization property.

2. Preliminaries

In this section, a brief review of the traditional wavelet neural networks and the NN-OSELM is presented.

2.1. Wavelet Network. The standard form of WN with one output y is

$$y = \sum_{i=1}^{m} w_i \psi_i \left(x_j, t_i, d_i \right) \quad t_i, d_i \in R^n, \ m \in N, \tag{1}$$

where $\psi_i(\cdot)$ refers to the activation function for the m hidden nodes where N represents the set of all integers. The wavelet activation functions can be any wavelet mother functions (i.e., Morlet, Mexican-hat, 1st Gaussian function derivative, etc.). The symbol w_i refers to the connection weights between the hidden nodes and the output node, while x_j is the applied input vector to the n input nodes $x_j = (x_1, x_2, \ldots, x_n)$. The multivariate wavelet mother function in the hidden nodes can be determined by the product of wavelet mother functions as follows:

$$\psi = \prod_{j=1}^{n} \psi_i \left(x, t_{ij}, d_{ij} \right) \qquad t_{ij} = \left(t_{i1}, t_{i2}, \ldots, t_{in} \right), \ d_{ij} = \left(d_{i1}, d_{i2}, \ldots, d_{in} \right). \tag{2}$$

The wavelet function parameters t_i and d_i are called translation and dilation (scale) parameters, respectively. The translation parameter determines the central position of the wavelet, whereas the dilation parameter controls the waves spread. These parameters can be defined from the data set available [14].

2.2. NN-OSELM. For single-layer feedforward neural networks with hidden nodes governed by additive sigmoid or RBF activation functions and based on ELM algorithm theories [20], an online sequential extreme learning machine (OSELM) is developed by [12] in a unified way to deal with industrial applications that the training data comes one by one or chunk by chunk. To explain the principles of the OSELM, suppose a SLFN of m hidden nodes and RBF activation function can be expressed as below,

$$y(x) = \sum_{i=1}^{m} w_i \Phi \left(\| x - c_i \| \right) \quad c_i \in R, \tag{3}$$

where $\Phi(\cdot)$ is a Gaussian type radial basis function and c_i is the center vector for neuron i and w_i is the output weights. Now, for a number of K input/output training samples that equals the number of SLFN hidden nodes (i.e., $K = m$), if the input weights, biases, and hidden node parameters are randomly assigned and independent from the training data sets, an analytic and simple way to calculate the estimated output weights $\hat{\omega}$ is applying least square solutions on the cost function as follows:

$$\mathcal{F}(\hat{\omega}) = \min \| H\hat{\omega} - Y \|, \tag{4}$$

where H is the SLFN hidden layer output and Y the real output. Here, the estimated output weights can be determined by inverting H with a single step where a zero training error can be realized. However, when the number of training samples is greater than the number of hidden nodes ($K > m$),

the estimated weights $\widehat{\omega}$ can be considered by using a pseudoinverse of H to give a small nonzero training error $|e| > 0$,

$$\widehat{\omega} = H^{\dagger}Y, \tag{5}$$

where $H^{\dagger}$ is pseudoinverse of H and the solution given by (5) can be rewritten as below [21],

$$\widehat{\omega} = \left(H^{T}H\right)^{-1}H^{T}Y. \tag{6}$$

Based on the above, the learning procedure of the OSELM algorithm consists of two phases, namely, the initial phase and the sequential learning phase. In initial phase, suppose a chunk of initial training samples $(u_{j_0}, y_{j_0}) \in \mathscr{R}^{K_0} : j_0 = 1, 2, \ldots, K_0$, where $K_0 \subseteq K$ is reached such that $m \leq K_0$; then the estimated weights can be found by

$$\widehat{\omega}_0 = (T_0)^{-1}H_0^{T}Y_0, \tag{7}$$

where

$$T_0 = \left(H_0^{T}H_0\right), \tag{8}$$

$$H_0 = \begin{bmatrix} \psi(X_1, t_1, d_1) & \cdots & \psi(X_1, t_m, d_m) \\ \vdots & \vdots & \vdots \\ \psi\left(X_{K_0}, t_1, d_1\right) & \cdots & \psi\left(X_{K_0}, t_m, d_m\right) \end{bmatrix}_{K_0 \times m},$$

$$Y_0 = \begin{bmatrix} y_1 \\ \vdots \\ y_{K_0} \end{bmatrix}^{T}_{K_0 \times 1}, \tag{9}$$

$$\widehat{\omega}_0 = \begin{bmatrix} w_1 \\ \vdots \\ w_m \end{bmatrix}^{T}_{m \times 1}$$

while X_j is the applied input vector to the n input nodes $X_j = (x_{j_1}, x_{j_2}, \ldots, x_{j_n})$. Now, for the sequential phase, suppose another chunk of data is reached (u_{j_1}, y_{j_1}), $j_1 = K_0 + 1, K_0 + 2, \ldots, K_0 + K_1$, and $m \leq K_1$; then the minimizing problem becomes

$$\mathscr{F}(\widehat{\omega}) = \min \left\| \begin{bmatrix} H_0 \\ H_1 \end{bmatrix} \widehat{\omega}_1 - \begin{bmatrix} Y_0 \\ Y_1 \end{bmatrix} \right\|, \tag{10}$$

where

$$H_1 = \begin{bmatrix} \psi\left(X_{K_0+1}, t_1, d_1\right) & \cdots & \psi\left(X_{K_0+1}, t_m, d_m\right) \\ \vdots & \vdots & \vdots \\ \psi\left(X_{K_0+K_1}, t_1, d_1\right) & \cdots & \psi\left(X_{K_0+K_1}, t_m, d_m\right) \end{bmatrix}_{K_1 \times m} \tag{11}$$

and then

$$\widehat{\omega}_1 = (T_1)^{-1}\begin{bmatrix} H_0 \\ H_1 \end{bmatrix}^{T}\begin{bmatrix} Y_0 \\ Y_1 \end{bmatrix}, \tag{12}$$

where

$$T_1 = \left(\begin{bmatrix} H_0 \\ H_1 \end{bmatrix}^{T}\begin{bmatrix} H_0 \\ H_1 \end{bmatrix}\right). \tag{13}$$

To express $\widehat{\omega}_1$ in terms of $\widehat{\omega}_0$, the detailed derivation formula can be found in [12], and it is described as

$$\widehat{\omega}_1 = \widehat{\omega}_0 + (T_1)^{-1}H_1^{T}\left(Y_1 - H_1\widehat{\omega}_0\right). \tag{14}$$

For the subsequenced chunks of data samples, the recursive least square solutions are applicable, and the previous arguments can be generalized for K_{s+1}. Suppose a data set $(u_{j_{s+1}}, y_{j_{s+1}})$, $j_{s+1} = K_s + 1, K_s + 2, \ldots, K_s + K_{s+1}$, where $s \geq 0$; then a recursive algorithm for updating T_{s+1} can be written as follows:

$$T_{s+1} = T_s + H_{s+1}^{T}H_{s+1} \tag{15}$$

and (14) will be

$$\widehat{\omega}_{s+1} = \widehat{\omega}_s + \left(T_{s+1}\right)^{-1}H_{s+1}^{T}\left(Y_{s+1} - H_{s+1}\widehat{\omega}_s\right). \tag{16}$$

Finally, the OSELM algorithm for SLFN can be summarized in the following steps:

(1) Initialize the network parameters (input weights, biases, and hidden nodes parameters) randomly.

(2) Determine T_0 and $\widehat{\omega}_0$ using (8) and (7) for the initial chunk of samples ($K_0 \geq m$).

(3) Set $s = 0$.

(4) Determine T_{s+1} and $\widehat{\omega}_{s+1}$ using (15) and (16) for the next sample data K_{s+1}.

(5) Set $s = s + 1$, $\widehat{\omega}_s = \widehat{\omega}_{s+1}$, and $T_s = T_{s+1}$.

(6) Go to Step (4) until all training data finish.

3. Nonlinear System Identification Using WN-OSELM

Many of nonlinear systems can be represented using the nonlinear autoregressive with exogenous inputs (NARX) model. Taking single-input-single-output systems as an example, this can be expressed by the following nonlinear difference equation:

$$\begin{aligned} y(k) &= f\left(y(k-1), \ldots, y\left(k-n_y\right), u(k), \ldots, u\left(k-n_u\right)\right) \\ &\quad + e(k), \end{aligned} \tag{17}$$

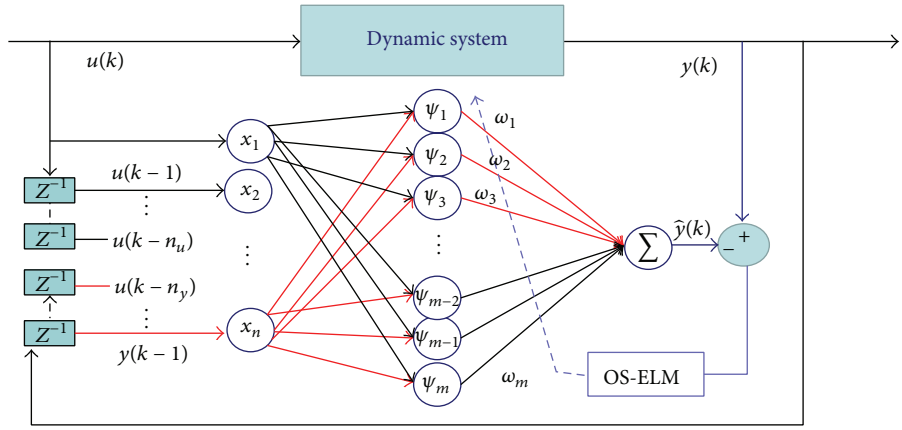

Figure 1: WN-OSELM model for nonlinear system identification.

where f is unknown nonlinear mapping functions, n_y is the number of lagged feedback predictor sequences $y(k)$ (output), n_u is the number of lagged feedforward predictors $u(k)$ (input), and $e(k)$ is the fitting residual assumed to be bounded and uncorrelated with the inputs. Several methods can be applied to realize the NARX model including polynomials, neural networks, and wavelet networks [22]. In this work, the proposed wavelet network with NARX called series-parallel structure is shown in Figure 1, where the real plant outputs are assumed to be available and can be used for prediction so that the stability of the network is guaranteed [23].

On the other hand, since the type of activation functions has an excessive impact on the network efficacy [24], the activation function of the WN hidden nodes is selected to be Morlet wavelet function as follows:

$$\psi(x_J, t_1, d_1) = \left\{\cos\left(5\left(\frac{x_j - t_i}{d_i}\right)\right)\right\} \exp\left(-0.5\left(\frac{x_j - t_i}{d_i}\right)^2\right) \quad (18)$$
$$i = 1, \ldots, m.$$

Morlet wavelet function has been introduced by the authors in [3, 9, 10] and showed better generalizing capability which is proved statistically in different applications, noting that the type of the activation function that has been used with WN-OSELM in [15] was first derivative of Gaussian function.

It is important to state that the OSELM algorithm for WN is similar to SLFN introduced in Section 2.2. The only difference is the initialization procedure of the input weights and the wavelet activation function parameters which is determined from the available data set. It is preferred to yield attention of the WN parameters to deliver a fine initial point to the OSELM algorithm for the training phase. Assigning wavelet activation function parameters, namely dilation and translation, is a critical issue; wavelet functions with a small dilation value are low frequency filters and may make some wavelets too local, whereas increasing dilation factors the wavelet behaves as high frequency filter and in both cases the components of the gradient of the cost function could be having an effect on speed of convergence and the accuracy of learning algorithm [25]. In this paper, the initializing of wavelet activation function parameters is done using density function and recursive algorithm [16] which is based on the initial input data that is available. Consider WN in the form of (1) with $j = 1, \ldots, n$ input nodes, and $i = 1, \ldots, m$ hidden nodes, and suppose chunks of input/output data set $(u_k, y_k) \in \mathcal{R}^K$ are available for training; then the input domain space can be defined by $[a_j, b_j]$ which holds the input item x_j in all observed samples. Now, consider the dilation parameter d_{j1} defined as the center of the parallelepiped; then $d_{j1} = 0.2(a_j - b_j)$. To initialize t_{j1}, a point p_j is selected between a_j and b_j so it divides interval $[a_j, b_j]$ into two parts. To find the point p_j, the estimated density functions from the available input output observations $[u, y]$ are considered such that the point p_j could be the center of gravity of domain $[a_j, b_j]$. In this method, to find the point p_j, the "density" functions estimated from noisy input/output observations $[u, f(u)]$ are used such that the point p_j is the center of gravity of $[a_j, b_j]$,

$$p_j = \int_a^b u\rho(u)\, du, \quad (19)$$

where

$$\rho(u) = \frac{|df(u)/du|}{\int_a^b |df(u)/du|\, du}. \quad (20)$$

Here, the assumptions that $p_j = (a_j + b_j)/2$ and $t_{j_1} = p_j$ are followed, the same as in [3]. Recursively, for each subinterval the same procedure applied for all n input nodes to get a form of $(d_{12}, t_{12}), \ldots, (d_{nm}, t_{nm})$ matrix. However, the number of data samples required to fill up appropriate matrix should be greater than the number of hidden nodes [12].

On the other hand, the WN input layer weights and biases are initialized using Nguyen Widrow approach that is explained in detail in [3]. This algorithm creates initial weight and bias values so that the active region of each neuron in the input layer is distributed evenly across the input space. Therefore, the training phase works faster because each area of the input space has neurons that covers the active region of the data set, and this is the main advantage of

Nguyen Widrow approach over purely randomizing weights and biases [17].

The initialization procedure of Nguyen Widrow approach [3] is as follows:

(1) Initialize small (random) input weights w_{old} in between [0.5, −0.5]: {the weights from input nodes n to hidden nodes m}.

(2) Compute $\beta = C \times m^{1/n}$ {C is a constant ≤ 0.7}.

(3) Compute $w_{\text{new}} = \beta \times w_{\text{old}}/\|w_{\text{old}}\|$ {normalized weights}.

(4) Initialize bias values b randomly between β and $-\beta$.

4. Simulation and Examples

The simulations are based on comparisons between WN-OSELM and NN-OSELM that applied to three nonlinear dynamic systems, namely, the magnetic levitation (MagLev), continuous stirred tank reactor (CSTR), and robot arm system. The comparison of this method to some conventional nonlinear dynamic systems identification methods will not be conducted here because a comparison between NN-OSELM and some conventional methods was covered in [12]. All the input output data sets and the simulations are carried out using MATLAB 7.11 environment running in an ordinary PC with 2.27 GHZ CPU. Moreover, the collected data sets of the plants are detailed in Table 1, where the first two rows of the table show the ranges of the amplitudes and widths of the input pulses.

The third and fourth rows indicate the outputs value each with corresponding units and the sampling time for each one. However, the halves of the total of each plant sample are used for training and the other half is used for testing and validation.

In addition, Table 2 defines the architecture of the WN and NN models with each plant, where the activation function of WN is Morlet wavelet function and for NN selected to be a sigmoid function [12]. For the NRAX structure, the input features and the number of the hidden nodes are different for each model because we seek here the best performance of both models by selecting the finest delayed input/output and the optimal hidden nodes numbers that give better performance to the models. The first two rows in Table 2 show the number of delayed inputs and delayed outputs (regressors) and both were determined using a heuristic approach called sequential forward search (SQFS) [18].

Moreover, the third row states the optimal number of hidden layer nodes based on final prediction criterion (FPEC) [19]. The last row of Table 2 yields the initial chunk of the training data samples which should be greater than the number of hidden nodes and has been determined by adding number 50 over the hidden nodes number. Note that any integer value equal to or larger than 0 can be used, just to make sure that the number of hidden nodes is equal to or less than the presented data samples in the initial stage only.

In order to compare the model output with the plant actual output, the relative error index [7] can be used to measure the performance of the identified model which is defined as

$$\overline{E} = \left(\frac{\sum_{k=1}^{T_{\text{test}}} \left| y(k) - \widehat{y}(k) \right|^2}{\sum_{k=1}^{T_{\text{test}}} \left| y(k) - \overline{y}(k) \right|^2} \right), \tag{21}$$

where $y(k)$ and $\widehat{y}(k)$ are the output measurements over the data set and associated for one-step-ahead predictions, respectively. T_{test} refers to the length of the validation data set and $\overline{y}(k)$ refers to the mean of the measured output over the validation data set and can be determined by

$$\overline{y}(k) = \frac{1}{T_{\text{test}}} \sum_{k=1}^{T_{\text{test}}} y(k). \tag{22}$$

Note that $\overline{E}$ is called also normalized error because the value of $\sum_{k=1}^{T_{\text{test}}} |y(k) - \overline{y}(k)|^2$ is considered as the standard deviation of the measured output and it is always constant for a given test data set; therefore the best model has minimum $\overline{E}$ with validation data test. However, to determine $\overline{E}$ in percentage form the formula below is applicable for performance test,

$$\text{PI} = \left(1 - \frac{\sum_{k=1}^{T_{\text{test}}} \left| y(k) - \widehat{y}(k) \right|^2}{\sum_{k=1}^{T_{\text{test}}} \left| y(k) - \overline{y}(k) \right|^2} \right) \times 100\%. \tag{23}$$

Furthermore, by comparing the PI for the simulated models output over the validation data set, the performance of the model can be evaluated where the best model has maximum PI with validation data.

Table 1: The data sets for the nonlinear plants.

	CSTR	MagLev	Robot arm
Input range	(0, 4)	(−1, 4)	(−15, 15)
Input interval (sec)	(5, 20)	(0.1, 1)	(0.1, 2)
Output range	(20, 23)	(0, 6.5)	(−3, 3)
Sample time (sec)	0.2	0.01	0.05
Total number of samples	8000	4000	1400

Table 2: Parameters of the models structures for the plants.

	CSTR		MagLev		Robot arm	
	NN	WN	NN	WN	NN	WN
Number of delayed inputs	1	2	2	1	1	2
Number of delayed outputs	2	2	2	3	2	2
Number of hidden nodes	7	7	9	14	7	19
Initial chunk of training data	57	57	59	64	57	69

4.1. Example 1: Magnetic Levitation System. The equation of motion for this system stated in [26] is

$$\frac{d^2 y(t)}{dt^2} = -g + \left(\frac{P}{M} \right) * \frac{i^2(t)}{y(t)} - \left(\frac{\beta}{M} \right) * \frac{dy(t)}{dt}, \tag{24}$$

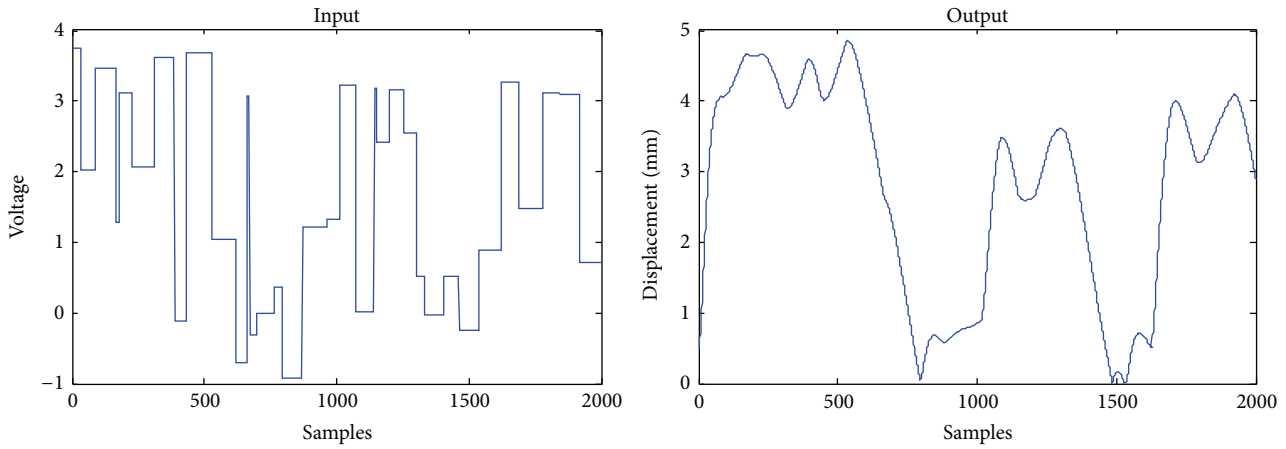

Figure 2: The input and output data set of magnetic levitation system.

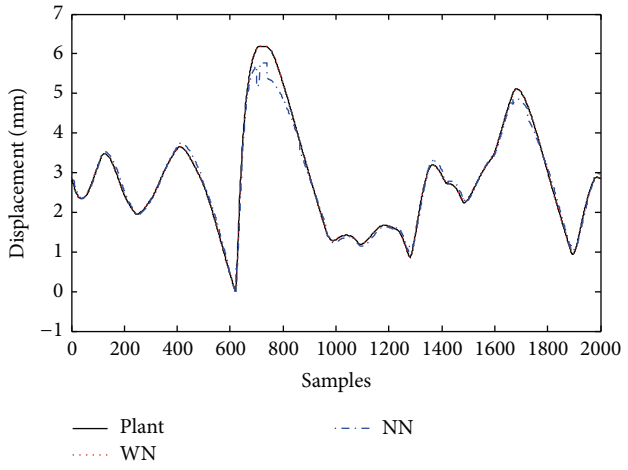

Figure 3: WN and NN outputs of the magnetic levitation system.

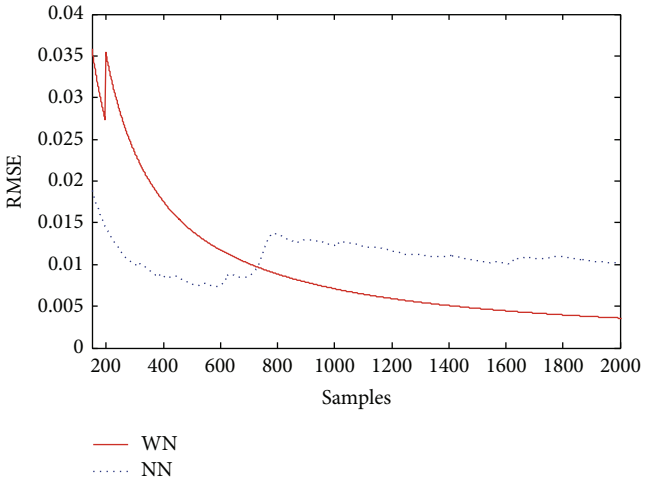

Figure 4: The RMSE curve of the NN and WN for MagLev.

where $x(t)$ is the displacement of the magnet ball, $i(t)$ is the current that controls the electromagnet flux, M is the mass of the magnet ball, and g is the gravitational constant. The factor β is a tacky friction parameter (damping constant), and θ is a field power constant. These parameters can be found in [26]. The magnetic levitation system data was collected at a sampling interval of 0.01 seconds to form the input and output time series data set shown in the Figure 2.

The simulations of WN and NN on the system are carried out and the results of the models are shown in Figure 3. The differences is notable and the PI performances of WN reached 97.50% which means the capability of capturing the plant dynamics is much more than NN that its PI reached 87.20%.

On the other hand, the root mean square error (RMSE) curves of the training phase for both models are shown in Figure 4. It is clear that the convergence capability of the WN over NN is much better in convergence than NN. Note that this system has fast dynamic response, yet WN was capable of extracting plant dynamics correctly due to the capability of the wavelet localization property by means of both translation and dilation parameters.

4.2. Example 2: Continuous Stirred Tank Reactor (CSTR). The dynamics of the CSTR system [27] can be described by

$$\frac{dl(t)}{dt} = F_1(t) + F_2(t) - 0.2 * \sqrt{l(t)},$$

$$\frac{dr(t)}{dt} = (C_1 - r(t)) * \frac{F_1}{l} + (C_2 - r(t)) * \frac{F_1}{l} - \frac{r(t)}{(1 + r(t))^2}, \quad (25)$$

where $l(t)$ is the fluid level, $r(t)$ is the concentration result at the output, $\theta_1(t)$ is the flow rate of first liquid feed source with concentration $C_1 = 24.9$, and $\theta_2(t)$ is the flow rate of the second feed with concentrations $C_2 = 0.1$. The CSRT data sets are obtained preparatory for both training and testing as shown in Figure 5, where each sample time t is considered as 0.2 seconds.

However, the simulation result displayed WN much better than NN as in Figure 6, where the PI was 91.66% for

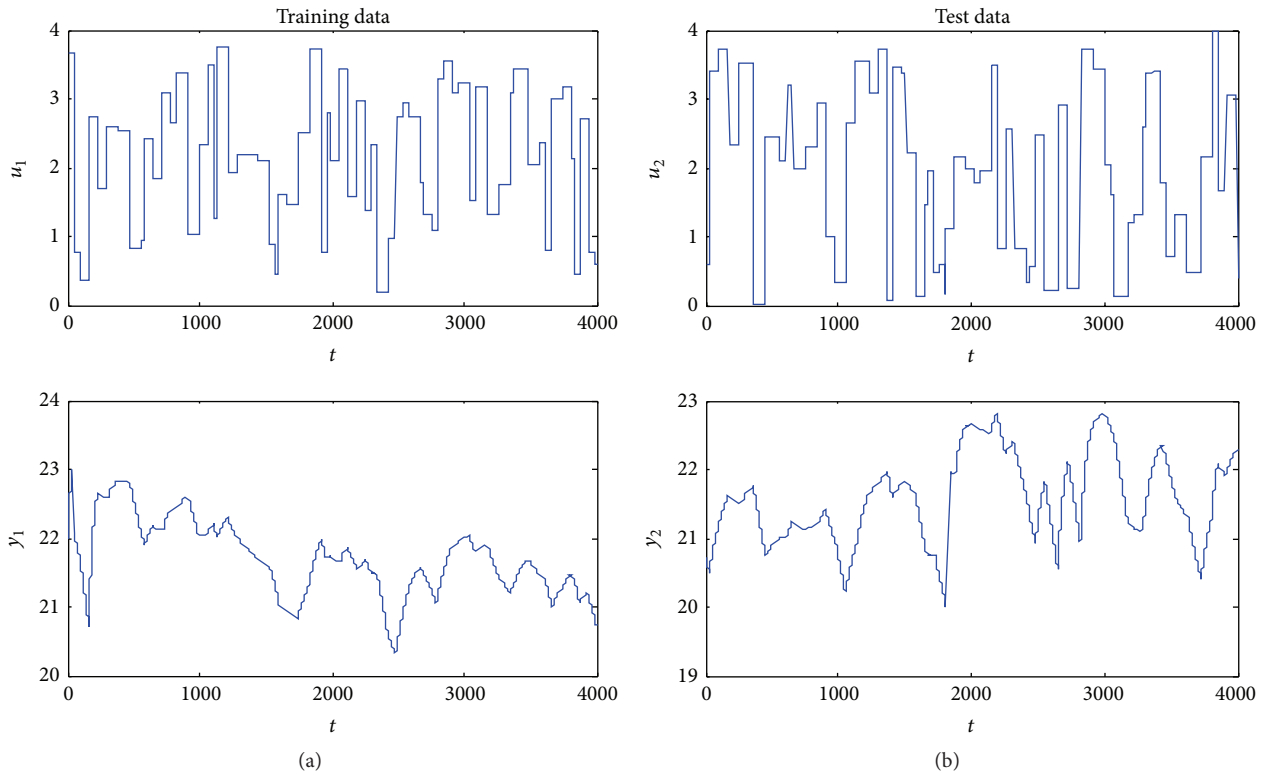

Figure 5: The CSTR trained input/output (a). Tested input/output (b).

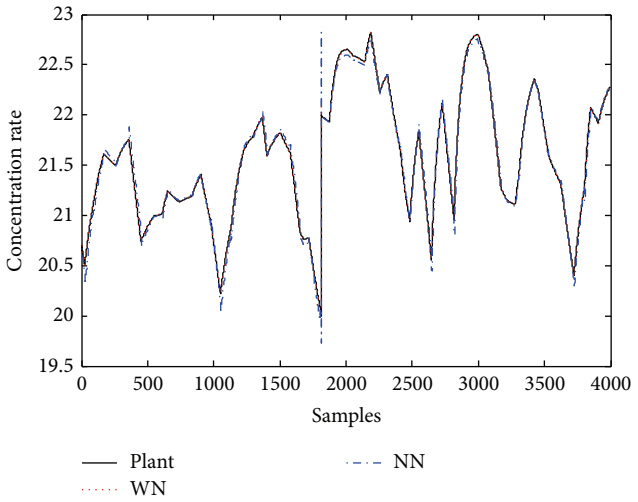

Figure 6: The WN and NN outputs for CSTR.

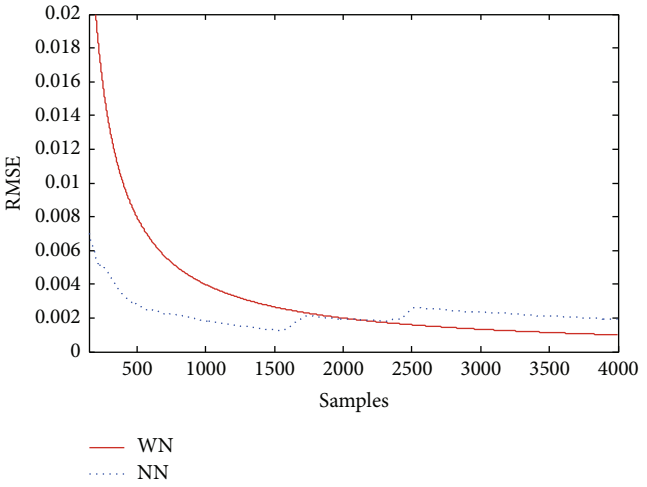

Figure 7: The RMSE of the NN and WN for CSTR.

WN while NN showed 80.13%. Note that NN suffers from sharp discontinuity ends, while WNs with decompositions properties showed were able to overcome the sharp discontinuities with better responses.

The learning evaluation in terms of RMSE for both NN and WN models of the CSTR can be shown in Figure 7. It is clear that the capability of WN is superior to that of NN in terms of fast convergence.

4.3. Example 3: Robot Arm. The system consists of a single-link robot arm that controls the movement of the arm [28]. The equation of motion dynamics is

$$\frac{d^2m(t)}{dt^2} = -10 * \sin(m(t)) - 2 * \frac{dm(t)}{dt} + T, \quad (26)$$

where $m(t)$ is the angle of the arm and T is the DC motor torque. To obtain input and output data sets, different torque

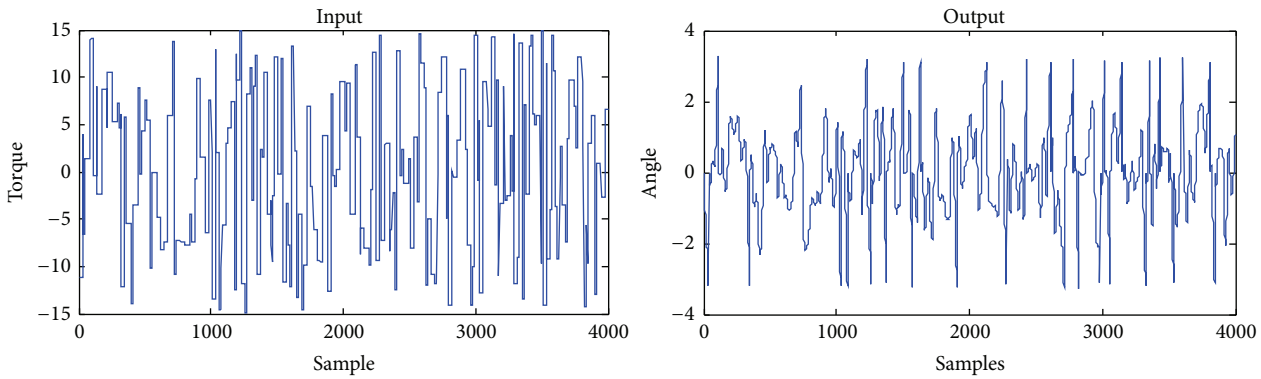

Figure 8: The input output data set of robot arm.

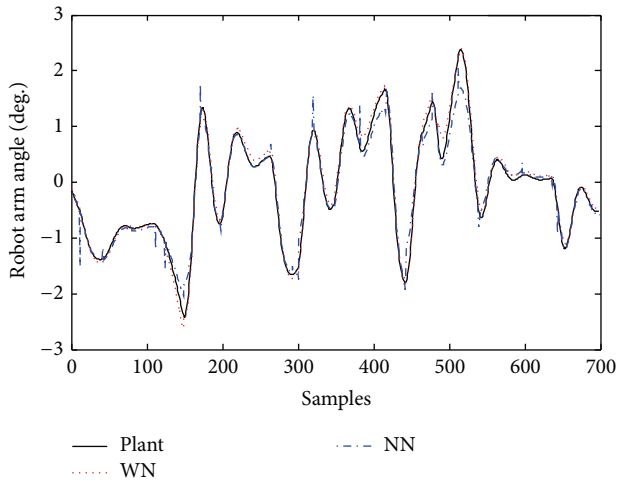

Figure 9: WN and NN outputs for robot arm.

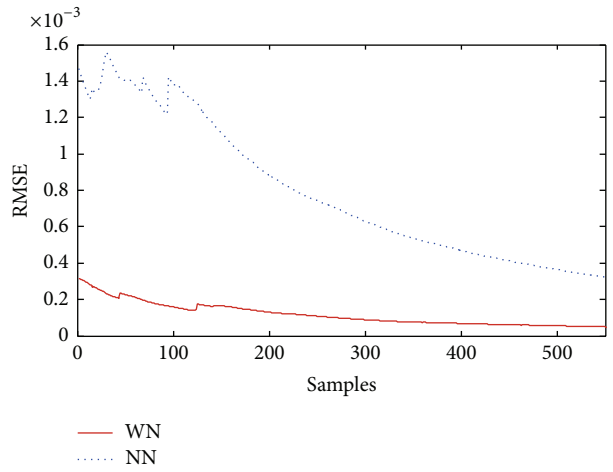

Figure 10: The RMSE of the NN and WN for robot arm.

levels are applied to get different arm angles of total 8000 samples and Figure 8 shows 4000 samples for the training data set for learning process.

However, the simulation results for robot arm system in Figure 9 showed that PI for WN still better than NN, where PI with WN reached 89.79% while NN was 79.37%.

On the other hand, the training RMSE curves are shown in Figure 10. For robot arm system, it is obvious that the NN curve declines gradually with nonsmooth convergence, while the WN curve was stable from the beginning and much better with RMSE value.

From all the three examples, it is clear that OSELM with WN has a jumping effect of convergence and much smoothness over NN and this can be explained in this way; since the sequential implementation of the least squares solution is applied here, then all the convergence results of recursive least squares (RLS) solutions can be applied also [12], and this means that the distribution of eigenvalues of H has a variable impact on the speed of convergence. In NN case, eliminating eigenvalue spread problem cannot be implemented due to lack of knowledge of H due to random initialization of H parameters which is not in the case of WN that the way of choosing its initial parameters is known. Therefore, a priori chosen initial parameters depending on the available data set will be very useful to enhance the OSELM convergence performance. Note that, in case of NN, the best of 50 trails (this is stated in [13]) are selected for the comparison process. Table 3 shows the elapsed time in seconds of the learning process for both WN and NN. In all simulation tests the WN showed faster learning process over NN, and this can be explained by the proper initializing of the WN parameters which enabled to converge faster even when WN both input and hidden node numbers was larger than NN as in case of robot arm system. However, it important to mention that the initialization process time of the WN parameters is higher than the NN initialization process time due to the recursive method of WN and random process of NN.

5. Conclusions

Wavelet network is proposed with OSELM to be used as nonlinear system identification scheme for control purposes in many industrial applications where sequential learning algorithms do not require retraining whenever new data is received. The initialization of wavelet network parameters is done using density function and recursive algorithm. The numbers of input and hidden nodes are selected using SQFS

Table 3: The WN and NN learning times and PI for the three systems.

	MagLev		CSRT		Robot arm	
	NN	WN	NN	WN	NN	WN
Time (seconds)	0.9204	0.5772	1.5288	0.9984	1.5132	1.1700
PI %	87.20	97.50	80.13	91.66	79.37	89.79

and FPEC, respectively; this is different from the approach introduced in [15] in which the numbers of inputs and hidden nodes were fixed. The simulations are carried out on three nonlinear dynamic systems and compared with SLFN-OSELM. From the results, it was found that the initialization procedure of the networks parameters plays a big role in the OSELM performance and consequently in the generated model performance. In other words, there was instability in the training phase of NN because of random initialization of the network parameters, while the training curve of WN was smooth and stable due to analytical initialization method of the wavelet mother function parameters. Moreover, the identified models by WN showed a better modeling accuracy over NN. The results overall showed the superiority of WN over NN in modelling performance and fast learning ability which realize that this work provides a very short learning time and better accuracy in many tests. In other words, the proposed algorithm can provide solutions for real-time varying system identification in some critical applications.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1–3, pp. 489–501, 2006.

[2] G.-B. Huang, D. H. Wang, and Y. Lan, "Extreme learning machines: a survey," *International Journal of Machine Learning and Cybernetics*, vol. 2, no. 2, pp. 107–122, 2011.

[3] K. Javed, R. Gouriveau, and N. Zerhouni, "SW-ELM: a summation wavelet extreme learning machine algorithm with a priori parameter initialization," *Neurocomputing*, vol. 123, pp. 299–307, 2014.

[4] G. Zhao, Z. Shen, C. Miao, and Z. Man, "On improving the conditioning of extreme learning machine: a linear case," in *Proceedings of the 7th International Conference on Information, Communications and Signal Processing (ICICS '09)*, Piscataway, NJ, USA, December 2009.

[5] N. Sureshbabu and J. A. Farrell, "Wavelet-based system identification for nonlinear control," *IEEE Transactions on Automatic Control*, vol. 44, no. 2, pp. 412–417, 1999.

[6] A. K. Alexandridis and A. D. Zapranis, "Wavelet neural networks: a practical guide," *Neural Networks*, vol. 42, no. 1, pp. 1–27, 2013.

[7] S. A. Billings and H.-L. Wei, "A new class of wavelet networks for nonlinear system identification," *IEEE Transactions on Neural Networks*, vol. 16, no. 4, pp. 862–874, 2005.

[8] Y. Oussar and G. Dreyfus, "Initialization by selection for wavelet network training," *Neurocomputing*, vol. 34, no. 4, pp. 131–143, 2000.

[9] J. Cao, Z. Lin, and G.-B. Huang, "Composite function wavelet neural networks with extreme learning machine," *Neurocomputing*, vol. 73, no. 7–9, pp. 1405–1416, 2010.

[10] J. Cao, Z. Lin, and G.-B. Huang, "Composite function wavelet neural networks with differential evolution and extreme learning machine," *Neural Processing Letters*, vol. 33, no. 3, pp. 251–265, 2011.

[11] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, 2004.

[12] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Transactions on Neural Networks*, vol. 17, no. 6, pp. 1411–1423, 2006.

[13] T. Blu and M. Unser, "Wavelets, fractals, and radial basis functions," *IEEE Transactions on Signal Processing*, vol. 50, no. 3, pp. 543–553, 2002.

[14] J. Zhang, G. G. Walter, Y. Miao, and W. N. W. Lee, "Wavelet neural networks for function learning," *IEEE Transactions on Signal Processing*, vol. 43, no. 6, pp. 1485–1497, 1995.

[15] D. M. Salih, S. B. M. Noor, M. H. Marhaban, and R. M. K. R. Ahmad, "Wavelet network based online sequential extreme learning machine for dynamic system modeling," in *Proceedings of the 9th Asian Control Conference (ASCC '13)*, pp. 1–5, Istanbul, Turkey, June 2013.

[16] B. Zhou, A. Shi, F. Cai, and Y. Zhang, "Wavelet neural networks for nonlinear time series analysis," in *Advances in Neural Networks—ISNN 2004*, vol. 3174 of *Lecture Notes in Computer Science*, pp. 430–435, Springer, 2004.

[17] D. Nguyen and B. Widrow, "Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '90)*, pp. 21–26, San Diego, Calif, USA, June 1990.

[18] J. Tikka and J. Hollmén, "Sequential input selection algorithm for long-term prediction of time series," *Neurocomputing*, vol. 71, no. 13–15, pp. 2604–2615, 2008.

[19] L. Ljung, *System Identification Theory for the User*, Prentice-Hall, Upper Saddle River, NJ, USA, 1999.

[20] G.-C. Liao, "Hybrid improved differential evolution and wavelet neural network with load forecasting problem of air conditioning," *International Journal of Electrical Power and Energy Systems*, vol. 61, no. 1, pp. 673–682, 2014.

[21] C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems*, Prentice-Hall, New York, NY, USA, 1974.

[22] H. L. Wei, S. A. Billings, and M. A. Balikhin, "Wavelet based non-parametric NARX models for nonlinear input-output system identification," *International Journal of Systems Science*, vol. 37, no. 15, pp. 1089–1096, 2006.

[23] S. Postalcioglu and Y. Becerikli, "Wavelet networks for nonlinear system modeling," *Neural Computing & Applications*, vol. 16, no. 4, pp. 433–441, 2007.

[24] G. Daqi and Y. Genxing, "Influences of variable scales and activation functions on the performances of multilayer feedforward neural networks," *Pattern Recognition*, vol. 36, no. 4, pp. 869–878, 2002.

[25] Y. Oussar and G. Dreyfus, "Initialization by selection for wavelet network training," *Neurocomputing*, vol. 34, pp. 131–143, 2000.

[26] H. Mikulascaron and M. Kamensky, "Constrained magnetic levitation control," in *Proceedings of the 16th IFAC World Congress*, Prague, Czech Republic, 2005.

[27] K. Ankit, B. Mohit, and N. V. Paras, "Implementation of neural model predictive control in continuous stirred tank reactor system," *International Journal of Scientific & Engineering Research*, vol. 4, no. 6, pp. 1989–1996, 2013.

[28] M. Vaezi and M. A. Nekouie, "Adaptive control of a robotic arm using neural networks based approach," *International Journal of Robotics and Automation*, vol. 1, no. 5, pp. 87–99, 2010.