Journal of Engineering Science and Technology
Vol. 15, No. 1 (2020) 709 - 727

A WAVELET FUNCTIONAL LINK NEURAL NETWORK CONTROLLER TRAINED BY A MODIFIED SINE COSINE ALGORITHM USING THE FEEDBACK ERROR LEARNING STRATEGY

OMAR FAROUQ LUTFY

Control and Systems Engineering Department,
University of Technology-Iraq
E-mail: 60157@uotechnology.edu.iq

Abstract

In this paper, a Wavelet Functional Link Neural Network (WFLNN) structure is proposed to comprise the intelligent part of the Feedback Error Learning (FEL) scheme alongside the Proportional, Integral and Derivative (PID) controller. This control structure is used for controlling nonlinear dynamical systems. In particular, the synergy of the wavelet theory and the Functional Link Neural Network (FLNN) suggested in this work results in a more powerful neural structure with better approximation ability compared to those of the individual constituents. To optimize the parameters of the WFLNN, the recently developed Sine Cosine Algorithm (SCA) is adopted with certain modifications to enhance the searching performance of the original algorithm. The proposed algorithm is termed the Modified Sine Cosine Algorithm (MSCA), which has demonstrated superior optimization performance compared to other methods, including the original SCA. By conducting an extensive set of performance tests, the proposed control approach has attained remarkable results in terms of control precision, generalization ability, disturbance rejection, and handling system parameter variations. Moreover, a comparative study has revealed the superiority of the WFLNN compared to other neural network types acting as the feedforward controllers within the framework of the FEL.

Keywords: Feedback error learning, Functional link neural network, PID control, Sine cosine algorithm, Wavelet neural network.

1. Introduction

Synthesizing effective control schemes is an indispensable requirement for handling the ever-increasing complexity of modern industrial systems. Such control schemes require the design of robust controllers that can cope with the nonlinear and the complex nature of systems and at the same time can deal with the unexpected variations both within the systems themselves and in their surrounding environments. Among several control methods, computational intelligence techniques, in particular Artificial Neural Networks (ANNs), have proved their efficiency in controlling highly complex and nonlinear systems.

In this context, Direct Inverse Control (DIC) represents an important ANN-based control approach that has gained significant attention among researchers. This approach requires the development of a model that accurately captures the system inverse dynamics to act as a Feedforward (FF) controller. Nevertheless, the main drawback of this control method is that the training process of the FF controller is not goal-directed since the control error is not considered. As a result, the developed FF controller is not guaranteed to achieve the desired control objective [1, 2].

In order to alleviate the above difficulty, another unique inverse modelling strategy was proposed by incorporating both classical and intelligent controllers in a unified control structure, which was called the Feedback Error Learning (FEL) scheme. In this scheme, the classical controller is utilized in the Feedback (FB) path for stabilization, while the intelligent controller is utilized in the Feedforward (FF) path to handle the complexity and the nonlinearity of the controlled system [3].

Ever since its first appearance, various control approaches, including linear and nonlinear ones, were suggested within the framework of FEL. For example, Muramatsu and Watanabe [4] proposed a linear adjustable transfer function, which can learn the inverse dynamics of the system and subsequently, can act as the FF controller based on the FEL. In another work, Rouhollahi et al. [5] utilized the FEL to control a basal ganglia model, which was linearized about the equilibrium point. Sugimoto and Imahayashi [6] proposed a tuning law to realize a linear FF controller using the pole placement approach.

It is worth noticing that all the above control methods were used to control linear systems. In practice, however, most real systems are inherently nonlinear to some extent. Thus, more efforts were exerted towards designing nonlinear FEL schemes. To this end, as a nonlinear control approach, the Sliding Mode Control (SMC) theory was exploited for training the FF controller to obtain the inverse system dynamics using the FEL strategy [7-10]. Nevertheless, it is well-known that SMC design requires the availability of accurate system models. Moreover, this method results in high-frequency oscillations, known as the chattering phenomenon, which adds extra difficulty in this design approach.

Other more commonly used training techniques are represented by the Gradient Descent (GD) methods, which were extensively considered for training various types of ANNs based on the FEL strategy. For instance, Kouhi et al. [11] designed an $H\infty$-based FB controller to replace the PD controller in the FEL structure. The output of this FB controller was then used to train a neuro-fuzzy system using the GD algorithm. However, in this approach, the design of the H∞ controller required a good knowledge of the system dynamics, the parameters uncertainty, and the

disturbance nature. Unfortunately, this knowledge is not always available, especially for complex nonlinear systems. Kambara et al. [12] proposed a FEL structure to control the model of a two-link arm with two joints and six muscles.

Using ANNs, the authors designed a FB controller, an inverse statics FF controller, and a forward dynamics model of the arm. Each of the above networks was trained by reinforcement learning, the FEL, and the Back-Propagation Algorithm (BPA), respectively. This control approach is characterized by its high complexity and the necessity to use three different training methods, resulting in a heavy computational loading. As an application of the FEL strategy for the load frequency control in the restructured power system, Sabahi et al. [3] utilized a FF controller represented by a type-2 fuzzy neural network.

This controller was trained by the BPA to minimize the output of a Proportional Derivative (PD) controller, which was tuned by the trial and error method. Eciolaza et al. [13] implemented a FEL structure using a piecewise bilinear model as the FF controller, which was trained by the GD method to minimize the output of a conventional FB controller. Following the same research direction, several FEL structures were realized by exploiting the GD optimization methods [14-18].

Based on the preceding discussion, it is apparent that almost all previous works were accomplished based on GD methods and very little attention was given to utilize Evolutionary Algorithms (EAs) in designing FEL control schemes. An exception is the control approach presented by Topalov et al. [19], in which, the Genetic Algorithm (GA) was employed in the FEL structure. However, even in this approach, the authors used the GD method, in particular the BPA, to fine-tune the final result achieved by the GA.

In this context, it is well established that GD methods suffer from several limitations given by the slowness in the convergence speed, the inability to escape local minima of the search space, and the difficulty in selecting the appropriate value for the learning rate [20]. To avoid these limitations, evolutionary algorithms can provide more efficient optimization results, since they are more likely to get away from local minima and obtain the global solution of the optimization problem. As a recently developed EA, the Sine Cosine Algorithm (SCA), which was proposed by Mirjalili [21], is utilized in the present work to optimize the parameters of both the FB and the FF controllers within the FEL scheme.

As another design issue regarding the FEL scheme, different types of ANNs were considered in the previous works, including the neuro-fuzzy network [7, 11], the Multilayer Perceptron (MLP) [12, 14, 15], the Radial Basis Function (RBF) [22], and the Cerebellar Model Articulation Controller (CMAC) [23].

Besides the above networks, there are other powerful ANN types, namely the Wavelet Neural Network (WNN) and the Functional Link Neural Network (FLNN). In particular, the WNNs have shown better approximation results compared to traditional neural networks [24, 25]. Moreover, as a computationally efficient network, the FLNN has been applied in different modelling and control applications [26]. However, despite the desirable features of both the WNN and the FLNN, they were not considered in designing FEL structures.

To enhance the network approximation ability, Hsu [24, 26] proposed to combine the WNN and the FLNN using a unified architecture. In particular, the outputs of two separate networks, namely a WNN and a FLNN, were integrated by

a single node to generate the final network output. The GD method was used to find the optimal values for the parameters of the resulting network. However, this network requires more adjustable parameters compared to those of its individual constituents, which adds extra computational burden to the overall structure.

In this work, unlike the above structure, the WNN and the FLNN are combined in a simpler architecture that involves only dilation and translation parameters. The proposed network, which is called the WFLNN, is trained to capture the system inverse dynamics, and consequently to act as a FF controller. Based on the FEL strategy, the training process is performed by directly minimizing the output of the Proportional, Integral, and Derivative (PID) controller during the control phase, and hence, no prior inverse modelling training is needed. To avoid the limitations of the widely used GD algorithms in the FEL design, the parameters of both the PID and the WFLNN controllers were optimized by a modified version of the recently developed SCA.

The results of comprehensive simulation tests and comparative studies evidently demonstrate the favourable control performance of the proposed control structure. The remaining sections of this article are organized as follows: Section 2 sheds some light on the FEL scheme. The details of the intelligent FF controller represented by the proposed WFLNN are provided in Section 3. Section 4 describes the original SCA together with its modified version, namely the MSCA. In Section 5, the results of several evaluation tests are exhibited and discussed. Finally, Section 6 gives the main conclusions from this work.

2. Feedback Error Learning using the WFLNN

As a biologically inspired learning technique, the Feedback Error Learning (FEL) is based on the idea that a FF controller can be trained to reproduce the inverse dynamics of the controlled system by minimizing the output of a FB controller. In most previous FEL applications, conventional and intelligent controllers were utilized in the FB and the FF loops, respectively. Figure 1 illustrates the FEL scheme adopted in this work.

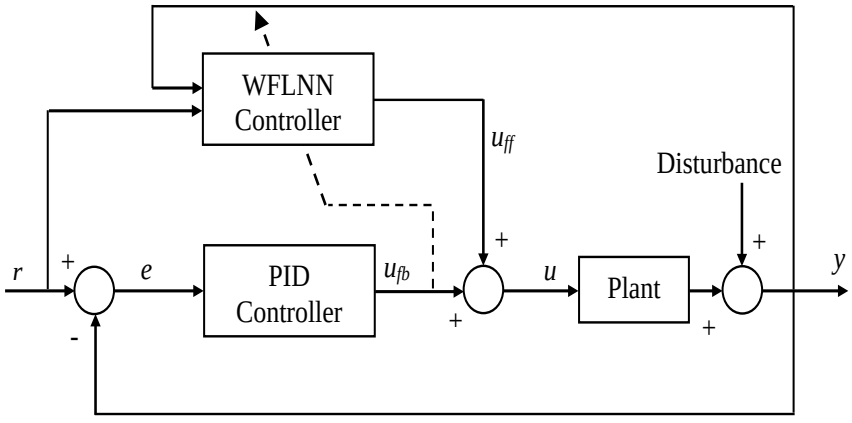


Fig. 1. A block diagram of FEL scheme, in which, the PID is used as FB controller and the WFLNN is used as FF controller.

The main control objective of the structure in Fig. 1 is to minimize the control error representing the difference between the reference signal and the actual system output. To achieve this objective, a PID controller is used in the FB path to stabilize the system, handle external disturbances, and at the same time to provide its output as a teaching signal for the FF controller [14, 17]. On the other hand, as it is evident in Fig. 1, the proposed WFLNN is used in the FF path of the structure to serve as an intelligent controller that can handle the nonlinearity of the controlled system. In this structure, both the PID and the WFLNN controllers work simultaneously to produce total control action. Hence, the total control signal, $u(k)$, is computed as the summation of the PID output, $u_{fb}(k)$, and the WFLNN output, $u_{ff}(k)$. For the FF controller training, the PID output, $u_{fb}(k)$, is utilized as a teaching signal to be minimized using a particular error criterion. In this work, this criterion was selected to be the Integral Square of Errors (ISE), which is given by the following equation [3, 11]:

$$J = \frac{1}{2}\sum_{k=1}^{N} e^2(k), \tag{1}$$

where $e(k) = u_{fb}(k) = u(k) - u_{ff}(k)$. Therefore, by minimizing the error signal in Eq. (1), the WFLNN is trained to attain the system inverse dynamics.

3. Wavelet Functional Link Neural Network

Out of several types available for ANNs, the Functional Link Neural Network (FLNN) is characterized by a special structure that uses functional links to produce the network output. FLNNs have gained widespread popularity as universal approximators to solve various nonlinear problems [26, 27]. To further enhance the approximation ability of the FLNN, a proposal was made in this work to exploit the good localization features of the wavelet functions in producing the final network output.

The resulting network, which is called the Wavelet Functional Link Neural Network (WFLNN), has demonstrated superior approximation accuracy compared to other networks, as will be shown in Section 5.5. In particular, the role of the wavelet transform in the proposed WFLNN is to enhance the approximation ability of the original FLNN by including an additional wavelet layer between the functional expansion stage and the output layer. In this regard, unlike conventional ANNs, the wavelet functions are spatially localized, and hence, they can offer more potential to enrich the input-output mapping relationship compared to conventional ANNs [24, 26]. Figure 2 illustrates the structure of the proposed WFLNN.

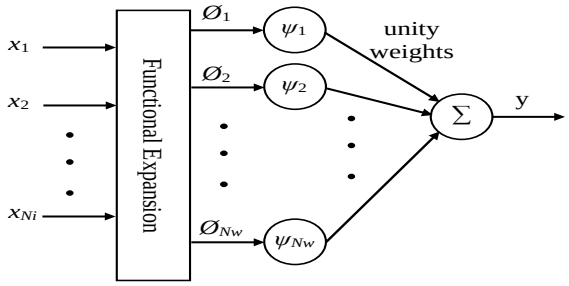


Fig. 2. Structure of WFLNN.

As it is evident from Fig. 2, the proposed WFLNN consists of a functional expansion stage, a wavelet layer, and an output layer. For the functional expansion stage, various functions can be used including power series, trigonometric, Chebyshev, and Legendre functions [24, 26, 27]. In particular, the trigonometric functions are adopted in this work, since they achieved the desired network accuracy. For *Ni* input variables, these functions can be described as follows [26]:

$$\varphi = [\varphi_1, \varphi_2, \dots, \varphi_{Nw}] = [x_1, cos(\pi x_1), sin(\pi x_1), \dots, x_{Ni}, cos(\pi x_{Ni}), sin(\pi x_{Ni})], \quad (2)$$

where *Ni* and *Nw* represent the number of input variables and the number of expansion terms, respectively. After the functional expansion stage, each resulting term enters the corresponding node in the wavelet layer, as shown in Fig. 2. Accordingly, the following operator is performed for each node in the wavelet layer:

$$z_j = d_j \varphi_j - t_j, \quad (3)$$

where j = 1, 2, …, *Nw*, d_j and t_j denote the dilation and the translation parameters of the jth node in the wavelet layer, respectively, and φ_j is j^{th} term from the functional expansion stage. To compute the output of each wavelet node, the Rational functions with Second-order Poles 1 (RASP1) wavelet function was utilized. This wavelet function is characterized by its rational form of strictly proper function and the utilization of simple double poles of second order. Therefore, the jth wavelet output can be determined according to the following RASP1 equation:

$$\psi_j(z_j) = \frac{z_j}{(z_j^2 + 1)^2}, \quad (4)$$

where z_j is the output from Eq. (3). As shown from Fig. 2, each wavelet output is transferred by a connection with a unity weight to the output node, which generates the final network output as the summation of all the received signals from the previous layer, as given by the following expression:

$$y = \sum_{j=1}^{Nw} \psi_j \quad (5)$$

On the basis of the above discussion, it can be noticed that there are only two sets of adjustable parameters in the proposed WFLNN structure, namely the dilation, d_j, and the translation, t_j, for each wavelet node. More specifically, these parameters are optimized based on minimizing the error criterion defined in Eq. (1) utilizing the proposed MSCA whose details are provided in the subsequent section.

4. Sine Cosine Algorithm

As a newly proposed evolutionary search method, the Sine Cosine Algorithm (SCA) is a stochastic population-based technique developed by Mirjalili [21] in 2016. This global optimization method is characterized by its simple procedure and the exploitation of a few control parameters, resulting in an efficient searching technique. The procedure of the SCA involves manipulating a set of candidate solutions for the problem being optimized by making these solutions fluctuate outwards or towards the best solution. This searching strategy is accomplished by making use of a mathematical model consisting of sine and cosine functions. In more details, the SCA employs the following position updating equations to perform the exploration and the exploitation stages [21]:

$$X_i^{t+1} = \begin{cases} X_i^t + r_1 \times sin(r_2) \times |r_3 P_i^t - X_i^t|, & r_4 < 0.5 \\ X_i^t + r_1 \times cos(r_2) \times |r_3 P_i^t - X_i^t|, & r_4 \geq 0.5 \end{cases} \quad (6)$$

where X_i^t represents the current solution position in the i^{th} dimension at the t^{th} iteration, r_1, r_2, r_3, and r_4 are random numbers, P_i is the destination point position in the i^{th} dimension and $||$ signifies the absolute value. In particular, the parameter r_1 is responsible for deciding the next position's movement direction, which might be either in the space between the solution and the destination point (when $r_1 < 1$) or outside it (when $r_1 > 1$). In this context, to guarantee a suitable balance between exploration and exploitation operators of the search space, the value of r_1 is updated adaptively according to the following equation:

$$r_1 = a - t\frac{a}{T}, \quad (7)$$

where t and T denote the current iteration and the maximum number of iterations, respectively, and a is a constant. On the other hand, the duty of parameter r_2 is to define how far the above movement, decided by r_1, should be either towards or outwards the destination solution. Specifically, r_2 takes random values from the interval $[0, 2\pi]$. The parameter r_3, which assigns a random weight for the destination solution P_i, is used to randomly emphasize ($r_3 > 1$) or deemphasize ($r_3 < 1$) the destination effect in setting the distance. Finally, the parameter r_4, which is a random number in [0, 1], uniformly selects either the sine or the cosine portions in Eq. (6) [21].

4.1. Modified sine cosine algorithm

Despite the attempt to incorporate both exploration and exploitations operations in the original algorithm, it has been shown that the SCA has the tendency to get stuck into sub-optimal regions of the search space. More specifically, this problem can be attributed to the insufficient exploration ability of the original SCA [28, 29]. Furthermore, from several evaluation tests, it was found that the original SCA did not provide the required variable diversity to effectively explore the search space, and hence, resulted in an insufficient accuracy in training the proposed WFLNN controller.

As a solution to this problem, two modifications were made in this work to boost the exploration ability of the original algorithm by introducing more highly fit solutions to the optimization procedure. The proposed algorithm was called the Modified Sine Cosine Algorithm (MSCA). In particular, as the first modification, the worst n solutions are replaced by new candidate solutions that are derived from the best solution attained so far. In this work, n was set to 20 solutions and the following formula was used to produce the candidate solutions:

$$x_{i,j}^{t+1} = P_j^t + \mu_{i,j}(x_{m1,j}^t - x_{m2,j}^t), \quad (8)$$

where i and j indicate the position and the dimension of x, respectively, P is the best solution obtained until now, $m1$ and $m2$ are two different integers that are randomly selected from 1 to the maximum number of solutions and at the same time they have to be different from the index of the current solution, i, and $\mu_{i,j}$ represents a random value selected from [-1, 1]. As a second modification, a random solution is generated at each iteration and its objective function is evaluated. If the objective function of this newly generated solution is worse than the worst solution, the worst solution is replaced by the destination point position. Otherwise, the worst solution

is replaced by the newly generated solution. It is worth mentioning that these two modifications significantly improved the searching ability of the original algorithm without requiring any additional control parameters to achieve the suggested steps. These features qualify the proposed MSCA to achieve better optimization results compared with other related methods, as will be shown in Section 5.4.

4.2. The utilized training procedure

The MSCA described above was employed according to the following steps:

Step 1: In this step, both the maximum number of iterations, *T*, and the number of solutions, *S*, in the MSCA are initialized.

Step 2: Form an initial set of *S* solutions by randomly generating each solution within certain bounds, as follows:

$$X_i^t = [x_{i,1}^t, x_{i,2}^t, \ \dots, x_{i,j}^t, \ \dots, \ x_{i,D}^t] \tag{9}$$

where *t* is the current iteration, $i = 1, 2, \dots, S$, $j = 1, 2, \dots, D$, *S* is the number of solutions in the population, *D* is the number of decision variables in each solution. More specifically, each one of these solutions represents a single controller.

Step 3: Evaluate the cost function for each solution, J_i^t. For this purpose, Eq. (1) is used as the cost function to perform the WFLNN training process.

Step 4: Update the best solution achieved thus far and assign it as the destination point, P^t, as follows:

$$P^t = \min_{i \in 1,2,\dots,S} J_i^t, \tag{10}$$

Step 5: Update the random parameters r_1, r_2, r_3, and r_4 as described in Section 4.

Step 6: Update each solution position utilizing Eq. (6).

Step 7: Sort the solutions based on their objective functions starting from the best solution to the worst one, as follows:

$$X_i^t = [X_{best}^t, \dots, X_{worst}^t] \tag{11}$$

Step 8: Replace the worst *n* solutions, in this work *n* was set to 20, by *n* newly generated solutions using Eq. (8), as follows:

$$X_i^t = X_{new}^t \tag{12}$$

where $i = S\text{-}n, S\text{-}n+1, \dots, S$ and $new = 1, 2, \dots, n$.

Step 9: Generate a solution, V^t, randomly within certain bounds and evaluate its objective function, J_V^t. If this objective function is worse than that of the worst solution, J_{worst}^t, the worst solution, X_{worst}^t, is replaced by the destination point position. Otherwise, the worst solution is replaced by the newly generated solution according to the following condition for the minimization problem under consideration:

$$X_{worst}^t = \begin{cases} P^t & if \ J_V^t \geq J_{worst}^t \\ V^t & if \ J_V^t < J_{worst}^t \end{cases} \tag{13}$$

Step 10: As the stopping condition, when the maximum number of iterations is achieved, the algorithm is terminated and the best solution attained so far is used as the final optimized controller parameters. Otherwise, go to Step 3. Similar to the training procedure illustrated above, the MSCA was applied for tuning the PID controller parameters using the following cost function:

$$J = \frac{1}{2}\sum_{k=1}^{N}(r(k) - y_p(k))^2, \quad (14)$$

where N is the number of samples, $r(k)$ and $y_p(k)$ are the desired and the actual system outputs, respectively.

5. Simulation Results

In this section, the applicability of the suggested control scheme to handle nonlinear systems is comprehensively assessed.

For optimizing the control system parameters, the training procedure of the MSCA described in the previous section was applied using 50 solutions (search agents) and 300 iterations for all the controlled systems.

These settings were adequate for achieving the desired control objectives. As the first design step to control each plant, the parameters of the PID controller were first obtained by the proposed MSCA.

In particular, the PID controller is described by the following discrete-time equation [30]:

$$u(k) = u(k-1) + K_p[e(k) - e(k-1)] + K_i e(k) + K_d[e(k) - 2e(k-1) + e(k-2)] \quad (15)$$

where K_p, K_i, and K_d are the proportional, integral, and derivative parameters, respectively. Subsequently, the trained PID controller is used to realize the FEL scheme shown in Fig. 1.

5.1. Control performance tests

The objective of these tests is to evaluate the control performance of the proposed scheme in handling the following nonlinear systems.

Plant 1

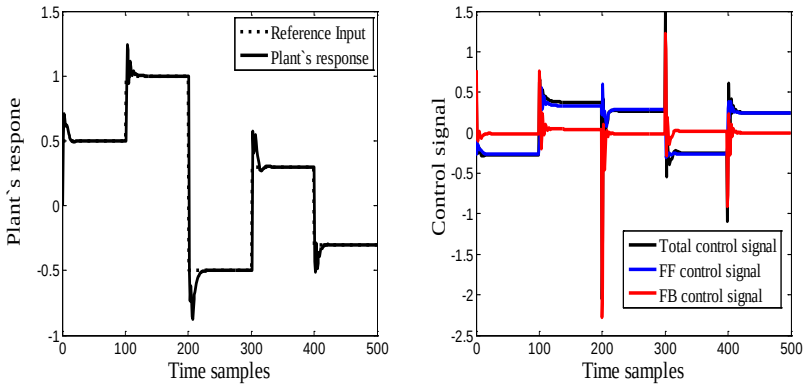
This plant model is represented by the following nonlinear discrete-time equation [31]:

$$x_1(k+1) = \frac{x_2^3(k) + x_2(k)}{1 + a_0 x_2^2(k)}, \quad x_2(k+1) = \frac{x_1(k) + x_2(k)}{1 + x_1^2(k)} + u(k), \quad y(k) = x_1(k) \quad (16)$$

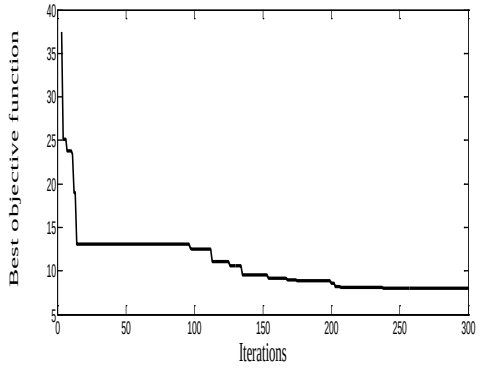
where $a_0 = 2$ in Eq. (16). To control this plant using the FEL scheme depicted in Fig. 1, the PID controller was first tuned by the proposed MSCA. In particular, the final tuned parameters for this controller were $K_p = 0.888$, $K_i = 0.211$, and $K_d = 0.448$.

With these settings, the PID controller output was used as a training signal to optimize the parameters of the proposed WFLNN utilizing the MSCA.

Figure 3 shows the simulation results for this plant.



(a) **(b)**



(c)

Fig. 3. Plant 1: (a) Plant's response, (b) Control signals, (c) Best cost function against iterations.

From Fig. 3(a), it is evident that the WFLNN-based FEL scheme has successfully achieved a good plant output response in tracking the desired changing step signal with some oscillations at the start of each signal change and with zero steady-state errors in all the signal parts. Figure 3(b) illustrates the control actions of the FF controller, the FB controller, and the total control signal represented as a blue line, a red line, and a black line, respectively. According to the FEL idea, training the FF controller dictates minimizing the FB controller output, which eventually results in making the total control signal approximately equal to the FF controller output. This behaviour is shown in Fig. 3(b), where it can be seen that the FB controller output, i.e., the red line, was minimized by the MSCA to achieve the desired control objective. This minimization is shown in Fig. 3(c), which demonstrates the best cost function, as defined in Eq. (1), against iterations.

Plant 2

As a complex nonlinear chemical system, a bioreactor process is considered to be controlled by the WFLNN-based FEL scheme. This bioreactor includes a tank that

contains water, nutrients, and biological cells. The nutrients and cells are mixed together inside the tank. To represent the dynamics of this process, the following continuous-time ordinary differential equations are used [32]:

$$\frac{dx_1}{dt} = -x_1 u + x_1(1 - x_2)e^{\frac{x_2}{\gamma}}, \quad \frac{dx_2}{dt} = -x_2 u + x_1(1 - x_2)e^{x_2/\gamma}\frac{(1+\beta)}{(1+\beta-x_2)} \tag{17}$$

As the states of this process, x_1 and x_2 represent the number of cells and the amount of nutrients, respectively. In Eq. (17), β is the growth rate and it is equal to 0.02, while γ is the nutrient inhibition parameter, which equals to 0.48.

In this process, the control objective is to keep the amount of cells at the desired level by removing the tank contents at a rate equals to the incoming flow rate, indicated by u in the process model. As initial conditions for the bioreactor states, the following settings were considered; $x_1 = 0.1207$ and $x_2 = 0.8801$. As the control objective, it is required to force the number of cells, x_1, to trace a trajectory defined by the following equation [32]:

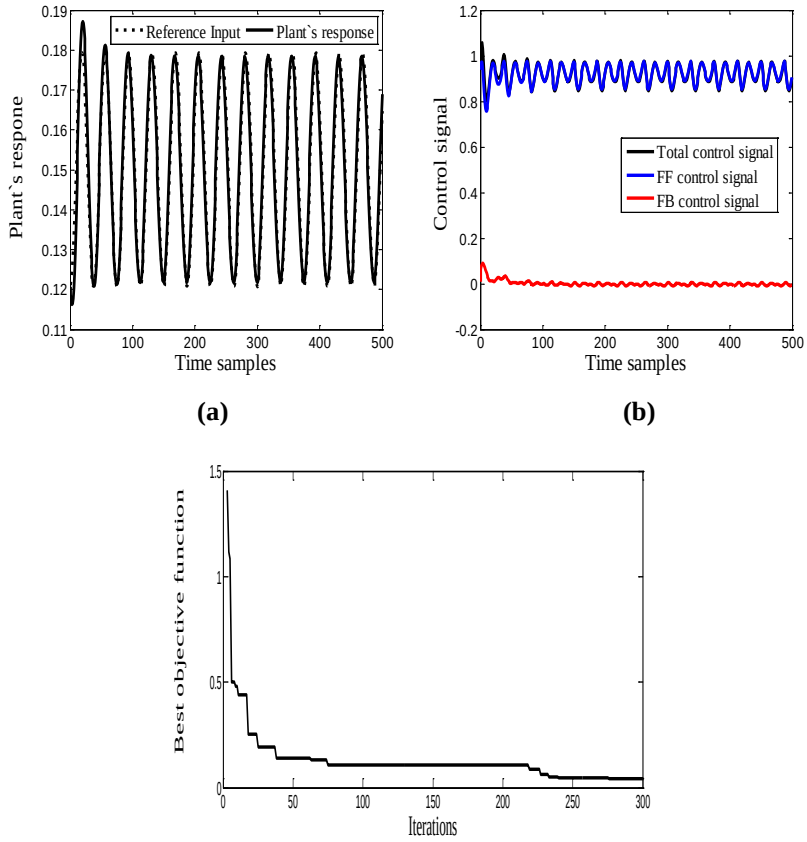
$$x_1(t) = 0.15 - 0.0293\, cos(\frac{8\pi t}{15}) \tag{18}$$

In the simulation, the fourth-order Runge-Kutta method was employed to numerically solve the bioreactor model of Eq. (17) with a step size of 0.1. Using the FEL scheme shown in Fig. 1, the PID controller was initially tuned by the MSCA resulting in the following parameters; $K_p = 1.136$, $K_i = 0.372$, and $K_d = 18.675$.

The tuned PID controller was then used as a FB controller providing its output as the teaching signal for the WFLNN training process. Figure 4 exhibits the output response, the control signals, and the optimization result for the bioreactor process. Specifically, Fig. 4(a) shows the accurate control result attained by the FEL control scheme, where the bioreactor response is almost identical to the desired trajectory of Eq. (18). In accordance with the FEL design concept, Fig. 4(b) obviously indicates that the FB controller output, specifically the red line, has been efficiently minimized to the zero levels by the MSCA, which makes the total control signal approximately equal to the FF controller output. The optimization performance of the MSCA is illustrated in Fig. 4(c), which shows the best cost functions against iterations.

5.2. Generalization tests

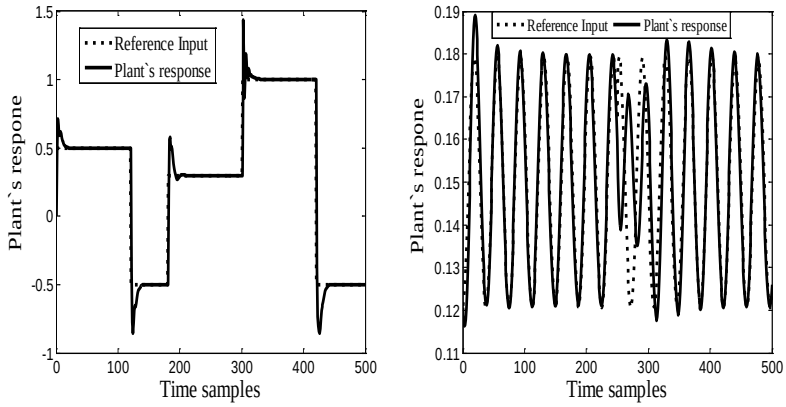
These tests aim at examining the generalization ability of the WFLNN controller in handling testing signals that differ from the signals used in the training stage. For this purpose, the same signals shown in Figs. 3(a) and 4(a) were used in the training stages of Plant 1 and Plant 2, respectively. On the other hand, different testing signals were used for the two plants. Specifically, a signal with different step changes than those of Fig. 3(a) was applied for Plant 1. As for Plant 2, the signal of Eq. (18) was applied for the first half of the simulation time, while for the other half, a signal similar to that of Eq. (18) (but with replacing the cosine by the sine function) was applied. Figure 5 illustrates the results of these generalization tests. Figure 5(a) shows that the WFLNN-based FEL scheme has successfully handled a testing signal for Plant 1 which is different from the one used in the training stage. The same remark applies for Plant 2, as shown in Fig. 5(b).



(a) **(b)**

(c)

Fig. 4. Plant 2: (a) Plant's response, (b) Control signals, (c) Best cost function against iterations.



(a) **(b)**

Fig. 5. Generalization tests for: (a) Plant 1 (b) Plant 2.

5.3. Robustness tests

These tests were conducted to assess the ability of the WFLNN-based FEL scheme to deal with unexpected conditions that the control system might encounter. Such conditions encompass both inherent variations in the system parameters and environmental changes represented by external disturbances. Each of these conditions is addressed in the following sections.

5.3.1. Parameter variation tests

For Plant 1, this test was made by changing the value of parameter a_0 in Eq. (16) during a specific interval. More precisely, a_0 was set to 5 from sample 301 to sample 375, while for the other samples a_0 was given the nominal value, which is 2. For Plant 2, two parameters, namely β and γ in Eq. (17), were both changed during the same interval from sample 134 to sample 138. During this interval, β and γ were set to 0.08 and 0.38, respectively, while for the remaining samples they were given the nominal values of 0.02 and 0.48, respectively. An important fact to notice here is that all the above changes were made during only the testing stage of the control system, which means that such changes were not accounted for in the training stage. The results of these parameter variation tests are shown in Fig. 6. Evidently, the control system has done well in accommodating unexpected changes in the system dynamics of Plants 1 and 2, as shown in Figs. 6(a) and (b), respectively.

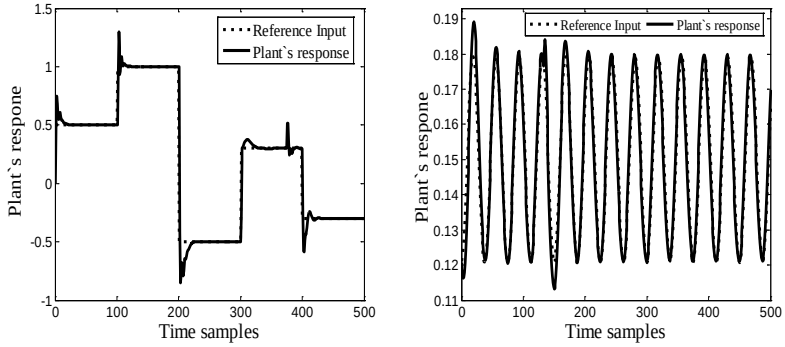
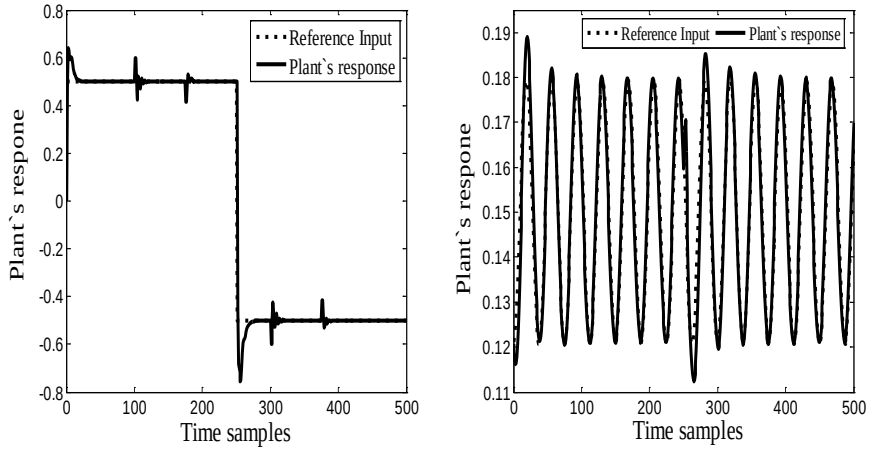


Fig. 6. Parameter variation tests for (a) Plant 1 (b) Plant 2.

5.3.2. Disturbance rejection tests

These tests objective is to analyse the control system robustness against unexpected external disturbances. For Plant 1, a disturbance with a magnitude of 20% of the plant output was injected for 75 samples during two periods, particularly during the periods $101 \leq k \leq 175$ and $301 \leq k \leq 375$. For Plant 2, a disturbance with a magnitude of 5% of the plant output was applied from sample 251 to sample 253. Similar to the parameter variation tests, all the disturbances were applied during only the control system testing stage, meaning that this system is not trained to manage the effects of these disturbances. Nonetheless, Fig. 7 clearly indicates the efficiency of the control system in this robustness test.

From all the above tests, which were conducted to thoroughly investigate the effectiveness of the proposed control approach, the WFLNN-based FEL has done well in controlling two different nonlinear systems. Specifically, in terms of control precision, the control system has resulted in good tracking responses to the desired signals for the two plants. On the other hand, in terms of generalization ability, the control system has adequately dealt with signals that were not encountered during the training stage. Finally, the tests in Section 5.3 have clearly revealed the robust performance of the WFLNN-based FEL scheme against unexpected changes caused by either the system itself or by its surroundings.



(a) Plant 1. **(b) Plant 2.**

Fig. 7. Disturbance rejection tests.

5.4. Comparison results of other optimization methods

To highlight the effectiveness of the proposed MSCA, a comparison analysis that involves other optimization techniques was conducted. These techniques include the Artificial Bee Colony (ABC), the Genetic Algorithm (GA), the Gravitational Search Algorithm (GSA), and the original SCA. The same plants defined in Section 5.1 were utilized to accomplish this study. To account for the stochastic nature of the considered optimization techniques and to achieve reliable comparison results, 10 runs were carried out for each technique and the average result was taken. The outcome of this comparative analysis is summarized in Table 1.

The term "FEL" in Table 1 refers to the final value of the feedback error learning cost function, as defined in Eq. (1), resulting from the training stage. On the other hand, "ISE" indicates the final integral square of errors, as defined in Eq. (14), resulting from the testing stage. Finally, "time" is the total processing time measured in seconds. From the results in Table 1, it is obvious that the proposed MSCA attained the least values for both the FEL and the ISE criteria for the two plants. In terms of processing time, the original SCA took slightly less time compared to the MSCA. More specifically, this difference in time was approximately 4 seconds for Plant 1 and 11 seconds for Plant 2. However, this small difference can be neglected in light of the superior results achieved by the MSCA.

Table 1. Comparison results of the ABC, the GA, the GSA, the original SCA, and the proposed MSCA as the training methods in the FEL scheme.

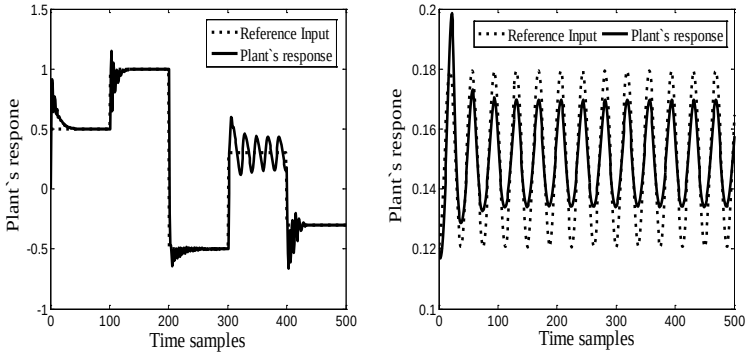
Optimization method	Criterion (average of 10 runs)	Controlled plant	
		Plant 1	**Plant 2**
ABC	FEL	8.030	0.059
	ISE	4.659	0.006
	Time (s)	127.047	181.223
GA	FEL	8.414	0.633
	ISE	4.644	0.074
	Time (s)	77.551	107.05
GSA	FEL	8.158	1.057
	ISE	4.564	0.107
	Time (s)	74.277	112.123
The original SCA	FEL	9.004	0.117
	ISE	4.710	0.018
	Time (s)	**69.068**	**103.839**
The proposed MSCA	FEL	**7.970**	**0.051**
	ISE	**4.528**	**0.004**
	Time (s)	72.969	115.091

5.5. Comparison results of other types of neural network controllers

The control result of the proposed WFLNN was compared with those of the MLP, the WNN, the RBF, and the original FLNN. Each of the above networks was trained by the MSCA to act as a FF controller in the FEL scheme. As was done before, 10 runs were made and the average result was adopted, as shown in Table 2. Obviously, the results in Table 2 signify that the proposed WFLNN controller performs the best of all the investigated neural controllers in terms of achieving the least values for the FEL and the ISE criteria for the two plants. Besides the results of Table 2 and to further exhibit the performance enhancement attained by the proposed WFLNN, Fig. 8 demonstrates the control performance of the original FLNN in controlling Plants 1 and 2, where it is obvious that the original FLNN has resulted in poor tracking responses for the two Plants. These results strongly suggest that the WFLNN is the best controller for the FEL scheme adopted in this work.

Table 2. Comparison results of the MLP, the WNN, the RBF, the original FLNN, and the proposed WFLNN.

Network type	Criterion (average of 10 runs)	Controlled plant	
		Plant 1	**Plant 2**
MLP	FEL	10.699	0.151
	ISE	5.223	0.027
WNN	FEL	8.127	0.166
	ISE	4.595	0.029
RBF	FEL	12.759	0.217
	ISE	4.536	0.034
The original FLNN	FEL	38.660	0.125
	ISE	4.994	0.021
The proposed WFLNN	FEL	**7.970**	**0.051**
	ISE	**4.528**	**0.004**



(a) Plant 1. **(b) Plant 2.**

Fig. 8. Control performance of the original FLNN for (a) Plant 1 (b) Plant 2.

6. Conclusions

A modified structure that combines the merits of both the wavelet theory and the FLNNs was proposed in this paper. This structure, which was called the WFLNN, was exploited as the intelligent part of a FEL scheme to control nonlinear systems. To accomplish the training stage, a modified version of the newly proposed SCA was employed. In particular, two modifications were made on the original SCA to enhance its searching performance. The resulting algorithm, which was termed the MSCA, was utilized to optimize the parameters of both the PID and the WFLNN controllers. The results of several assessment tests demonstrated the efficiency of the WFLNN-based FEL scheme with regards to control accuracy, generalization ability, and robustness against unexpected system parameter variations and external disturbances. Compared to other ANN controllers, the WFLNN achieved superior control results. Moreover, the MSCA resulted in favourable optimization performance compared with other techniques, including the original one. For future works, the proposed control scheme will be utilized to control real nonlinear systems.

Nomenclatures

a	A constant in Eq. (7)
a_0	A constant in Eq. (16)
d_j	Dilation parameter of the j^{th} node in the wavelet layer
$e(k)$	Error signal in Eq. (1)
K_p, K_i and K_d	Gains of the PID controller
$m1$ and $m2$	Two different integers that are randomly selected in Eq. (8)
N	Number of samples in Eq. (14)
Ni	Number of input variables to the WFLNN in Fig. 2
Nw	Number of expansion terms for the WFLNN in Fig. 2
P_i	Destination point position in the ith dimension in Eq. (6)
$r(k)$	Desired output in Eq. (14)
r_1, r_2, r_3, and r_4	Random numbers in Eq. (6)
T	Maximum number of iterations in Eq. (7)
t	The current iteration in Eq. (7)
t_j	Translation parameter of the j^{th} node in the wavelet layer

$u(k)$	Total control signal
$u_{fb}(k)$	PID output
$u_{ff}(k)$	WFLNN output
x_1	Number of cells in Eq. (17)
x_2	Amount of nutrients in Eq. (17)
X_i^t	Current solution position in i^{th} dimension in t^{th} iteration
y	Output of the WFLNN in Fig. 2
$y_p(k)$	Actual system output in Eq. (14)

Greek Symbols

β	Growth rate in Eq. (17)
γ	Nutrient inhibition parameter in Eq. (17)
$\mu_i j$	A random value selected from [-1, 1] in Eq. (8)
ϕ_j	j^{th} term from the functional expansion stage in Fig. 2
ψ_j	j^{th} wavelet output in Fig. 2

Abbreviations

ABC	Artificial Bee Colony
ANNs	Artificial Neural Networks
BPA	Back-Propagation Algorithm
CMAC	Cerebellar Model Articulation Controller
DIC	Direct Inverse Control
EAs	Evolutionary Algorithms
FB	Feedback
FEL	Feedback Error Learning
FF	Feedforward
FLNN	Functional Link Neural Network
GA	Genetic Algorithm
GD	Gradient Descent
GSA	Gravitational Search Algorithm
ISE	Integral Square of Errors
MLP	Multilayer Perceptron
MSCA	Modified Sine Cosine Algorithm
PD	Proportional Derivative
PID	Proportional, Integral and Derivative
RBF	Radial Basis Function
SCA	Sine Cosine Algorithm
SMC	Sliding Mode Control
WFLNN	Wavelet Functional Link Neural Network
WNN	Wavelet Neural Network

References

1. Denaï, M.A.; Palis, F.; and Zeghbib, A. (2007). Modeling and control of non-linear systems using soft computing techniques. *Applied Soft Computing*, 7(3), 728-738.
2. Antonelli, M.; Duran, A.J.; Chinellato, E.; and del Pobil, A.P. (2015). Learning the visual–oculomotor transformation: Effects on saccade control and space representation. *Robotics and Autonomous Systems*, 71, 13-22.

3. Sabahi, K.; Ghaemi, S.; and Pezeshki, S. (2014). Application of type-2 fuzzy logic system for load frequency control using feedback error learning approaches. *Applied Soft Computing*, 21, 1-11.
4. Muramatsu, E.; and Watanabe, K. (2004). Feedback error learning control without recourse to positive realness. *IEEE Transactions on Automatic Control*, 49(10), 1762-1770.
5. Rouhollahi, K.; Andani, M.E.; Karbassi, S.M.; and Izadi, I. (2017). Design of robust adaptive controller and feedback error learning for rehabilitation in Parkinson's disease: A simulation study. *IET Systems Biology*, 11(1), 19-29.
6. Sugimoto, K.; and Imahayashi, W. (2017). Direct tuning in feedback error learning control and its generalization to non-minimum phase plant. *IFAC-PapersOnLine*, 50(1), 5326-5331.
7. Kayacan, E.; Kayacan, E.; Ramon, H.; and Saeys, W. (2012). Neuro-fuzzy control with a novel training method based-on sliding mode control theory: Application to tractor dynamics. *IFAC Proceedings Volumes*, 45(22), 889-894.
8. Khanesar, M.A.; Kayacan, E.; Reyhanoglu, M.; and Kaynak, O. (2015). Feedback error learning control of magnetic satellites using type-2 fuzzy neural networks with elliptic membership functions. *IEEE Transactions on Cybernetics*, 45(4), 858-868.
9. Kayacan, E.; and Khanesar, M.A. (2016). Recurrent interval type-2 fuzzy control of 2-DOF helicopter with finite time training algorithm. *IFAC-PapersOnLine*, 49(13), 293-299.
10. Kayacan, E.; Peschel, J.M.; and Chowdhary, G. (2017). A self-learning disturbance observer for nonlinear systems in feedback-error learning scheme. *Engineering Applications of Artificial Intelligence*, 62, 276-285.
11. Kouhi, Y.; Adlgostar, R.; and Nourzadeh, H. (2007). Robust and FEL control design for MIMO flow-level control plant. *IFAC Proceedings Volumes*, 40(18), 535-540.
12. Kambara, H.; Kim, K.; Shin, D.; Sato, M.; and Koike, Y. (2009). Learning and generation of goal-directed arm reaching from scratch. *Neural Networks*, 22(4), 348-361.
13. Eciolaza, L.; Taniguchi, T.; Sugeno, M.; Filev, D.; Wang, Y.; and Michelini, J. (2014). PB model-based FEL and its application to driving pattern learning. *IFAC Proceedings Volumes*, 47(3), 7982-7987.
14. Kurosawa, K.; Futami, R.; Watanabe, T.; and Hoshimiya, N. (2005). Joint angle control by FES using a feedback error learning controller. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 13(3), 359-371.
15. Ruan, X.; Ding, M.; Gong, D.; and Qiao, J. (2007). On-line adaptive control for inverted pendulum balancing based on feedback-error-learning. *Neurocomputing*, 70(4-6), 770-776.
16. Neto, A.D.A.; Góes, L.C.S.; and Nascimento Jr., C.L. (2010). Accumulative learning using multiple ANN for flexible link control. *IEEE Transactions on Aerospace and Electronic Systems*, 46(2), 508-524.
17. Taheri, A.; Shoorehdeli, M.A.; Bahrami, H.; and Fatehi, M.H. (2014). Implementation and control of X-Y pedestal using dual-drive technique and feedback error learning for LEO satellite tracking. *IEEE Transactions on Control Systems Technology*, 22(4), 1646-1657.

18. Nguyen, T.N.; Su, S.; Celler, B.; and Nguyen, H. (2014). Advanced portable remote monitoring system for the regulation of treadmill running exercises. *Artificial Intelligence in Medicine*, 61(2), 119-126.
19. Topalov, A.V.; Kim, J.-H.; and Proychev, T.P. (1998). Fuzzy-net control of non-holonomic mobile robot using evolutionary feedback-error-learning. *Robotics and Autonomous Systems*, 23(3), 187-200.
20. Chakravarty, S.; and Dash, P.K. (2012). A PSO based integrated functional link net and interval type-2 fuzzy logic system for predicting stock market indices. *Applied Soft Computing*, 12(2), 931-941.
21. Mirjalili, S. (2016). SCA: A sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems*, 96, 120-133.
22. Yoon, M.; Lee, K.; and Sunwoo, M. (2007). A method for combustion phasing control using cylinder pressure measurement in a CRDI diesel engine. *Mechatronics*, 17(9), 469-479.
23. Zhang, X.; Eguchi, M.; and Ohmori, H. (2018). Diesel engine combustion control based on cerebellar model articulation controller (CMAC) in feedback error learning. *IFAC-PapersOnLine*, 51(31), 516-521.
24. Hsu, C.-F. (2013). A self-evolving functional-linked wavelet neural network for control applications. *Applied Soft Computing*, 13(11), 4392-4402.
25. Lutfy, O.F. (2014). Wavelet neural network model reference adaptive control trained by a modified artificial immune algorithm to control nonlinear systems. *Arabian Journal for Science and Engineering*, 39(6), 4737-4751.
26. Hsu, C.-F. (2013). Adaptive neural complementary sliding-mode control via functional-linked wavelet neural network. *Engineering Applications of Artificial Intelligence*, 26(4), 1221-1229.
27. Gotmare, A.; Patidar, R.; and George, N.V. (2015). Nonlinear system identification using a cuckoo search optimized adaptive Hammerstein model. *Expert Systems with Applications*, 42(5), 2538-2546.
28. Elaziz, M.A.; Oliva, D.; and Xiong, S. (2017). An improved opposition-based sine cosine algorithm for global optimization. *Expert Systems with Applications*, 90, 484-500.
29. Nenavath, H.; Jatoth, R.K.; and Das, S. (2018). A synergy of the sine-cosine algorithm and particle swarm optimizer for improved global optimization and object tracking. *Swarm and Evolutionary Computation*, 43, 1-30.
30. Lin, C-J. (2009). Nonlinear systems control using self-constructing wavelet networks. *Applied Soft Computing*, 9(1), 71-79.
31. Canelon, J.I.; Shieh, L.S.; and Karayiannis, N.B. (2005). A new approach for neural control of nonlinear discrete dynamic systems. *Information Sciences*, 174(3-4), 177-196.
32. Oysal, Y.; Becerikli, Y.; and Konar, A.F. (2006). Modified descend curvature based fixed form fuzzy optimal control of nonlinear dynamical systems. *Computers and Chemical Engineering*, 30(5), 878-888.