ISSN 1330-3651 (Print), ISSN 1848-6339 (Online) https://doi.org/10.17559/TV-20161025220853
Original scientific paper

Neuro-Controller Design by Using the Multifeedback Layer Neural Network and the Particle Swarm Optimization

Ramazan COBAN, Inayet Ozge AKSU

Abstract: In the present study, a novel neuro-controller is suggested for hard disk drive (HDD) systems in addition to nonlinear dynamic systems using the Multifeedback-Layer Neural Network (MFLNN) proposed in recent years. In neuro-controller design problems, since the derivative based train methods such as the back-propagation and Levenberg-Marquart (LM) methods necessitate the reference values of the neural network's output or Jacobian of the dynamic system for the duration of the train, the connection weights of the MFLNN employed in the present work are updated using the Particle Swarm Optimization (PSO) algorithm that does not need such information. The PSO method is improved by some alterations to augment the performance of the standard PSO. First of all, this MFLNN-PSO controller is applied to different nonlinear dynamical systems. Afterwards, the proposed method is applied to a HDD as a real system. Simulation results demonstrate the effectiveness of the proposed controller on the control of dynamic and HDD systems.

Keywords: hard disk drive; neuro-control; particle swarm optimization; recurrent neural networks

1 INTRODUCTION

In recent years, special attention has been devoted to neural network methodologies for model and control of nonlinear dynamic systems in various areas [1-4]. Recurrent fuzzy neural networks were successfully employed in control and model of dynamic systems in [5-8]. Some kinds of Recurrent Neural Networks (RNNs) were implemented into nonlinear dynamic systems for the purposes of control or model in [9-11].

Various computational optimization methods can be utilized to train artificial neural network controllers. The Genetic Algorithm (GA) in these optimization methods is an important approach preferred to train controllers. A Takagi–Sugeno–Kang based recurrent fuzzy neural network trained by the GA was suggested for control of nonlinear dynamical systems by Juang [12]. The Particle Swarm Optimization (PSO) method is faster, simpler, less complex in algorithm, and has superior performance than GA [13]. In [14], the PSO based fuzzy neuro-control was utilized for the ball and plate system. In the paper, the PSO was employed to optimize the fuzzy neural network connection weights. Sheikhan, Hemmati, and Shahnazi [15] proposed a neural network based intelligent controller for adaptive queue in transmission control protocol communication. In that research, the particle swarm optimization was utilized for optimization of the connection weights of the neural network controller. Devising the radial basis function networks by using the PSO method was considered by Tsekouras [16]. The PSO was used as a training algorithm for the conventional feed forward neural networks which is one of the static artificial neural networks in [17, 18]. The results obtained in these research works were compared with those achieved by using gradient-based algorithm.

Performance of the RNNs on account of their intrinsic recurrence and dynamic mapping attributes is superior to feedforward neural networks and radial basis function networks to design nonlinear controllers. Savran [19] demonstrated that the Multifeedback Layer Neural Network (MFLNN) among recurrent neural networks has more learning capability than some others in dynamic system identification and control. The MFLNN in the present paper is chosen to design a neuro-controller due to the fact that it has nonlinear architecture, successful convergent property, and quick training capability. Aksu and Coban [20] demonstrated that the PSO is a powerful training algorithm for the MFLNN.

Hard disk drives (HDDs) are the information storage devices and are utilised in numerous fields such as in computer systems and mobile communication engineering. In past few years, the track density of HDDs has been intensified to grow the storage volume. Hence, Reader Head (RH) should be properly positioned on the desired track location and relocated on another. Due to this significant motivation, the position control of the RH has attracted much attention of many researchers for its important applications. Particularly in mobile platform, locating the Reader/Writer Head (R/WH) on a proper place is an essential problem due to peripheral shockwaves and vibrations. The above mentioned condition decreases the tracking performance of HDDs. Furthermore, greater track density in HDDs demands greater precision of tracking performance in numerous computer-based platforms. The reduction of tracking error of the R/WH is a major task for engineers and scientists in the design of HDDs [21]. To achieve the previously specified goals, more effective and efficient controllers are required to retrieve the information successfully and accurately. A few techniques were put forward to enhance the hard disk performance in [22-25].

2 DESIGN METHODS

The MFLNN [19] plays an essential role in the artificial RNNs. The MFLNN includes four hidden connection layers. These hidden connection layers are composed of three feedback connection layers and one feedforward connection layer. It comprises an input connection layer and an output connection layer, too. Architecture of the MFLNN is illustrated in Fig. 1. Two locally connected and one globally connected feedback layers are located in the MFLNN architecture [19].The input and output signals of this architecture are represented by $x(k)$ and $y(k)$, respectively. W_1 is the weight parameter between the input and the hidden

connection layers, while W_2 is the weight parameter between the hidden and the output connection layers in the feedforward path. W_1^b, W_2^b and W_3^b designate the input weight parameters of the feedback connection layer. W_1^c, W_2^c and W_3^c designate the output weight parameters of the feedback connection layers. $h^c(k)$, $y^c(k)$ and $z^c(k)$ are the outputs of the feedback connection layer units. One-step time delay is indicated by z^{-1}. The bias connection weights are not demonstrated in the figure for easiness. Initial values of the hidden connection layer output (h) and the output connection layer output (y) are set to zero ($h(0)=0$, $y(0)=0$). The excitation signals of the activation functions for feedback connection layer units are [19]

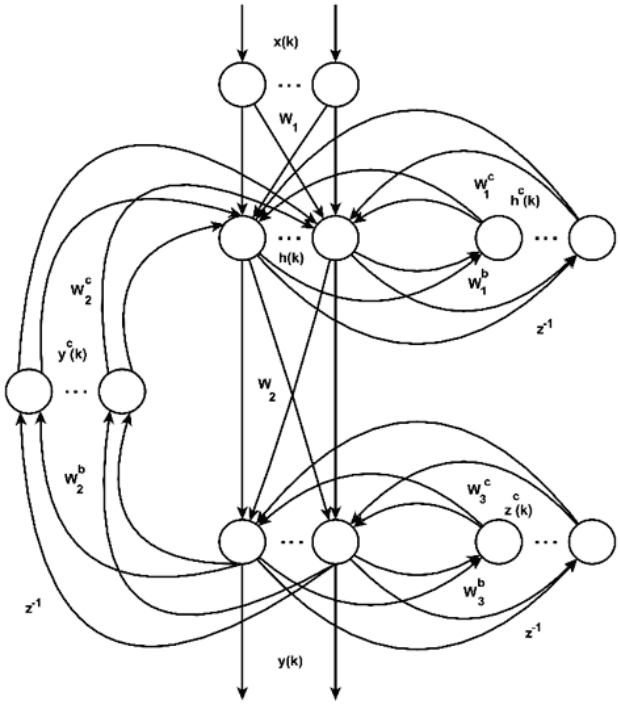


Figure 1 Architecture of the MFLNN

$$net_h^c(k) = [W_1^b h(k-1)] + B_1^b, \quad (1)$$
$$net_y^c(k) = [W_2^b y(k-1)] + B_2^b, \quad (2)$$
$$net_z^c(k) = [W_3^b y(k-1)] + B_3^b, \quad (3)$$

where B_1^b, $B_2^b$and B_3^b are the bias values of the feedback connection layer units. The output signals of the feedback connection layer units are [19]

$$h^c(k) = \varphi_h^c(net_h^c(k)), \quad (4)$$
$$y^c(k) = \varphi_y^c(net_y^c(k)), \quad (5)$$
$$z^c(k) = \varphi_z^c(net_z^c(k)), \quad (6)$$

where φ_h^c, φ_y^c, and φ_z^c denote the activations of the feedback layer units. $net_h(k)$ and $h(k)$ are evaluated by [19]

$$net_h(k) = [W_1 x(k)] + [W_1^c h^c(k)] + [W_2^c y^c(k)] + B_1, \quad (7)$$

$$h(k) = \varphi_h(net_h(k)), \quad (8)$$

where B_1 is the bias for hidden layer unit, and φ_h activations of the hidden layer, $net_y(k)$ is the local field of the output connection layer units. The output of the MFLNN y is calculated by [19]:

$$net_y(k) = [W_2 h(k)] + [W_3^c z^c(k)] + B_2, \quad (9)$$
$$y(k) = \varphi_y(net_y(k)), \quad (10)$$

The input signal propagates through the network in a forward direction from Eq. (1) through Eq. (10).

The PSO method is suggested by Kennedy and Eberhart [26]. Consider the *pbest* and *gbest* which represent the best place of each particle and that of neighbourhood of each particle. The movement of particles in search space is as follows [26, 27]:

$$v_{id} = wv_{id} + c_1 r_1(p_{id} - x_{id}) + c_2 r_2(p_{gd} - x_{id}), \quad (11)$$
$$x_{id} = x_{id} + v_{id}, \quad (12)$$

where d and i stand for the index of the search domain dimension and the particle in the swarm, respectively. The random numbers r_1 and r_2 are uniform distribution over the interval [0, 1]. The symbols c_1 and c_2 are positive coefficients. The notations w, v_{id}, and x_{id} are the inertia value, particle velocity, and position, respectively. p_{id} and p_{gd} denote the position of the *pbest* and *gbest,* respectively. The values of the c_1 and c_2 are calculated by [13]:

$$c_1 = c_{1max} - \frac{(c_{1max} - c_{1min})k}{K}, \quad (13)$$
$$c_2 = c_{1min} + c_{1max} - c_1, \quad (14)$$

where c_{1min} and c_{1max} are the lower and upper values of the coefficient c_1, respectively. The notations k and K are the epoch number and entire epochs, respectively. In the current study, the values of the parameters c_1 and c_2 are chosen as 1,5 and 2,5, respectively, according to trial and errors. Linearly decreasing inertia weight given by Eq. (15) is used [13]:

$$w_k = w_{max} - \frac{(w_{max} - w_{min})k}{K}, \quad (15)$$

where w_{max} is the maximum of the inertia weights, w_{min} is the minimum of the inertia weights. The parameters w_{max} and w_{min} are selected 0,9 and 0,4, respectively. In many research works, the population size or the total number of particles is preferred to a number between 10 and 70. In this study, the population size is set 60 upon a few trials. The amount of neighbourhoods is selected 50%.

In the present work, the standard PSO is enhanced. The enhancements are as follows: The particle which has the worst position is searched. Then it is altered with the swarm's best solution throughout the updating. The coefficient c_3 is inserted into Eq. (12) for updating the particle location given by

$$x_{id} = x_{id} + c_3 \times v_{id}. \quad (16)$$

The value of the coefficient c_3 is arbitrarily selected between 0,1 and 0,15. Eq. (16) is an improved form of Eq. (12). The mutation operator is included in the improved PSO method. One of the major disadvantages of heuristic methods is that the particles are trapped in local optima in

the search space. To solve this problem, the mutation operator is applied to the particles during the search. The uniform mutation with a predefined ratio is preferred. According to the predefined ratio, it adds a uniform random value within the range of the parameter to the value of the chosen particle. The values of the position and velocity parameters in the mutation process are changed according to the mutation rate. Therefore, the mutation process affects both the position and velocity parameters of the particles. In the present research, the mutation rate is augmented by the increment of 0,0001 in each iteration to prevent the algorithm trapping into the local best solution. In addition, on the mutation phase both the position and velocity parameters are exposed to the mutation operation. When the mutation rate is supplied, the position and velocity parameters of the particles are enlarged by an arbitrary value with a uniform distribution over an interval of ($KR\times[x_{min}, x_{max}]$), where KR denotes a positive coefficient. The key steps of the improved PSO are given in Fig. 2.

```
Set the parameters of the PSO
//Initialization
For Each particle in the swarm
Initialize position of particles randomly in the range [ X_min , X_max ]
Initialize velocity of particles randomly in the range [ V_min , V_max ]
End
//Iteration
Do
    For Each particle in the swarm
        Evaluate the fitness value
        If the fitness value is better than the best fitness value ( pbest ) in history
            Set present value as the new pbest .
    End
    Select the particle with the best fitness value in the neighborhood as the lbest
    For Each particle in the swarm
        Find the particle with the worst fitness value and change it
        to the particle with the best fitness value
        Compute particle velocity according to Eq. (11)
        Clamp the velocity in each dimension to the range [ V_min , V_max ] × KR
        Update position of particle according to Eq. (16)
        Clamp the position in each dimension to the range [ X_min , X_max ] × KR
        Apply mutation
    End
While Termination criteria is not fulfilled.
```

Figure 2 The pseudo code of the PSO method

In order to make evident the accomplishment of enhancements in the improved PSO, the well-known Easom function in literature given by Eq. (17) is employed:

$$f(\phi_1, \phi_2) = -\cos(\phi_1)\cos(\phi_2)e^{-(\phi_1-\pi)^2-(\phi_2-\pi)^2}, \qquad (17)$$

where $-100 \leq \phi_i \leq 100$ for $i = 1, 2$ The desired result is achieved in the 7th run for 5000 epochs by using the standard PSO method. However, the same result is achieved in the 49th epoch at the first run via the improved PSO method. This optimization study exhibits an error of 1×10^{-5}. In order to make a proper evaluation of the proposed improvements in the PSO, multiple runs on Easom function are needed. After 50 runs, the standard PSO and the improved PSO result in arithmetical mean errors of $1{,}3\times10^{-5}$ and $1{,}1\times10^{-7}$ for 5000 iterations, respectively.

The neural network architecture is important for the performance and adjustment of the dynamic system parameters. The MFLNN is preferred for the problem under consideration since the MFLNN that is a nonlinear model based on RNNs approximates not only linear but also nonlinear dynamic systems. For that reason, the controller architecture that is considered in this research is nonlinear. The block diagrams in Figs. 3a and 3b show the neuro-controller architecture in train and test stages, respectively. As seen from the figure the desired and past signals of the process in hand are the input signals to the neuro-controller. After the output signal of the neuro-controller is amplified, it comes into the process. Here, the architecture similar to one trained by the GA in [12] is used to control the nonlinear dynamic systems.

Update of the neural network connection weights plays an important role to find underlying mappings related to the system's behaviour. Numerous approaches have been proposed over the last few decades to optimize the neural networks weights. These approaches can be separated into two classes: gradient and non-gradient based approaches. A great number of results have been presented in the literature. The gradient-based approaches are back propagation, least squares, and Kalman filter. Some of the gradient-free methods are GA, PSO, and Evolutionary Programming (EP). The network connection weights are commonly trained by gradient-based approaches for dynamic system identification. In identification tasks there is not only actual output information but also desired one of the network. Therefore, the error signal between these outputs can be straightforwardly employed for error back-propagation. On the other hand, in control tasks, no knowledge about the reference value of the artificial neural network output exists. If so, it may be not economical to get such knowledge. The gradient-based techniques need such desired values of the artificial neural network's output or Jacobian of the dynamic system throughout the update. Thus, the gradient-based techniques cannot be utilized for neuro-controller design [25]. Because of this reason, the PSO method is preferred to adapt the MFLNN. The fitness value that is necessary for the PSO to train the MFLNN is acquired from the difference between the process output and desired signals.

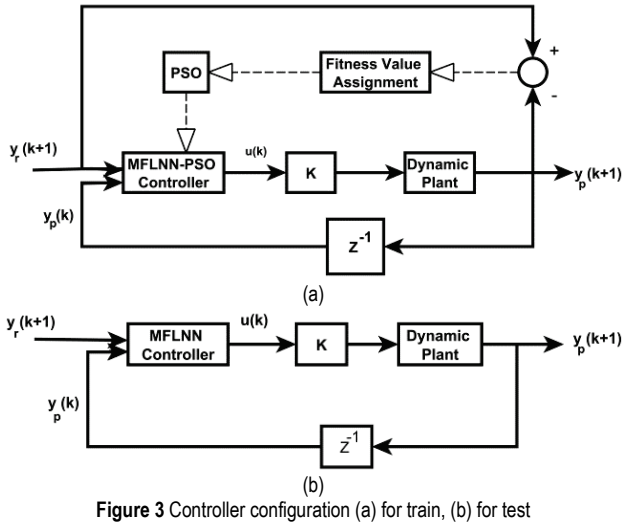


Figure 3 Controller configuration (a) for train, (b) for test

The error value has to be calculated for the fitness or cost function computations in the improved particle swarm optimization method. The Root Mean Square Error (RMSE) as a cost function is computed by

$$RMSE = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(y_r(k) - (y_p(k))^2}, \quad (18)$$

where N is number of the data set utilized in the train phase. y_p and y_r are the actual output of the control system and reference output, respectively.

At the beginning of each epoch, the initial values of parameters which are weights and bias values of the MFLNN are assigned randomly in the range [−1, 1]. As epoch carries on, this range is extended by a suitable number (KR) because the network weights take on greater values over the interval of [−10, 10] and the error convergence of the MFLNN is not guaranteed throughout the train. Keeping the positions and velocities of the PSO method over the interval of (KR×[−1, 1]) empowers the proposed method to converge in a fixed number of epochs. The train procedure lasts until a predefined termination criterion is fulfilled while the network weights are updated to minimize the fitness function. If no convergence is satisfied, the iteration stops and initializes with new random network weights and bias values.

The stability of the closed-loop control system in design of a neuro-controller is related to the convergence of the controller parameters during learning phase. A learning rule such as the PSO method updates the parameters (connection weights) of a neuro-controller from starting any initial conditions if the tracking error given by Eq. (18) converges to zero. So as to guarantee the stability and good performance of the closed-loop control system, the tracking error between the process output signal and the reference signal should converge to zero. Therefore, the closed-loop control system stability can be guaranteed by convergence.

Table 1 Characteristic parameters of the disk drive reader [30]

Parameter	Typical value
J	1 N m s^2/rad
b	20 N m s/rad
R	1 Ω
K_m	5 N m/A
L	1 mH

3 HARD DISK DRIVE

The head-positioning is a complex and significant closed-loop control system that is responsible for repositioning on each track in the least possible error and time. Both the track-seeking and the track-following are the two essential tasks for HDDs. Track seeking enables the R/WH to transport from the existing track to reference one in the short time as far as possible. Track-following makes sure that the R/WH is relocated precisely over a certain track with the smallest tracking error despite noise while data is read from or written to the disk [28-32]. The transfer function of the HDD servo control system is [30]

$$G(s) = \frac{K_m}{s(Js+b)(Ls+R)}, \quad (19)$$

where J is the inertia, b is the friction. R is the armature resistance, L is the armature inductance and K_m is the motor coefficient value. The values of these parameters are presented in Tab. 1. The simplified transfer function of the open-loop head reader servo system is achieved by

$$G(s) = \frac{Y(s)}{V(s)} = \frac{5000}{s(s+20)(s+1000)}. \quad (20)$$

If Eq. (20) and the zero-order hold for 0,1 second sampling time are employed, the difference equation is calculated by

$$y(k) = 1{,}1353 \times y(k-1) - 0{,}1353 \times y(k-2) + 0{,}0140 \times u(k-1) + 7{,}64 \times 10^{-3} \times u(k-2) + 6{,}9049 \times 10^{-7} \times u(k-3). \quad (21)$$

3 SIMULATION RESULTS

In this section, the MFLNN-PSO is applied to control some linear and nonlinear dynamic systems. Control architecture of the MFLNN-PSO for train and test is illustrated in Fig. 3a and Fig. 3b, respectively. The input signals to the MFLNN-PSO controller consist of present desired and past output signals of the system. Update of the proposed MFLNN-PSO controller for different linear and nonlinear dynamic systems and its behaviour on the HDDservo system are tested by means of different examples. In these examples, three hidden layer units are employed for the MFLNN. Four hidden connection layer units are also used to show effect of the number of the hidden connection layer units. Two input and one output connection layer units constitute the MFLNN. All the figures shown in the paper are drawn using the architecture consisting of three hidden connection layer units. Linear activation function in the input connection layer and hyperbolic tangent activation function in the output and hidden connection layers are chosen for the MFLNN to delineate both linear and nonlinear behaviour of the dynamic system. The PSO parameters employed in the current work are presented in Tab. 2.

Table 2 The PSO parameters

Parameters	Symbols	Values
Number of neighbours	NN	30
Population size	$S\ S$	60
Number of parameters	D	62
Minimum of confidence coefficients	c_{1min}, c_{2min}	1,5
Maximum of confidence coefficients	c_{1max}, c_{2max}	2,5
Minimum of third confidence coefficient	c_{3min}	0,1
Maximum value of third confidence coefficient	c_{3max}	0,15
Keep range value	KR	10
Mutation rate	PM	0,001
Minimum inertia	w_{min}	0,4
Maximum inertia	w_{max}	0,9

First, two different nonlinear dynamic system control tasks are studied. The results obtained in the given examples are compared with those in the Takagi–Sugeno–Kang-type Recurrent Fuzzy Network with Genetic learning (TRFN-G) [12] whose structure is the same as the MFLNN-PSO controller. In the example 1 and the example 2, the desired signals reported in [12] are employed. Moreover, the same desired signals are applied to HDD servo control system in example 3 and example

4. In the simulation examples, the controller gain K is fixed to 10.

Example 1: In the first numerical experiment a nonlinear dynamical system including three past outputs is employed in [12]:

$$y_p(k+1) = \frac{0{,}6y_p(k)+y_p(k-1)(y_p(k)+2{,}5)}{1+y_p^2(k)y_p^2(k-1)} + u(k), \quad (22)$$

where y_p is the output signal of the process. To update the neuro-controller, 250 data are acquired from the following equation:

$$\begin{aligned} &y_r(k+1) = 0{,}6y_r(k) + 0{,}2y_r(k-1) \\ &+0{,}6\sin(2\pi k/25), 1 \le k \le 110 \\ &= 0{,}6y_r(k) + 0{,}2y_r(k-1) + 0{,}2\sin(2\pi k/25) \\ &+0{,}4\sin(\pi k/32), 110 < k \le 250 \end{aligned} \quad (23)$$

where y_r is the reference. The desired signal utilized for test performance of the nonlinear dynamic system is acquired in the following.

$$\begin{aligned} &y_r(k+1) = 0{,}6y_r(k) + 0{,}2y_r(k-1) \\ &+0{,}2\sin(2\pi k/25) + 0{,}4\sin(\pi k/32), \\ &250 < k \le 500. \end{aligned} \quad (24)$$

Train is carried on for 9000 time steps. The cost function value utilized for the improved PSO method in train stage is computed by

$$RMSE = \sqrt{\frac{1}{250}\sum_{k=1}^{250}(y_p(k+1) - (y_r(k+1))^2}. \quad (25)$$

Table 3 RMSEs of the MFLNN-PSO and TRFN-G in Example 1

	TRFN-G	MFLNN-PSO	
RMSE_Train	0,0631	0,009315	0,008703
RMSE_Test	0,0536	0,007931	0,007544
ParameterSize	48	42	62

The RMSE values achieved in the results of the train and test phases are given in Tab. 3. The desired and process outputs are illustrated in Fig. 4. The result obtained by the train data is demonstrated in Fig. 4a, while that by the test data is shown in Fig. 4b. It looks that the MFLNN-PSO has a performance superior to that of the TRFN-G that includes more hidden connection layers. This means that the MFLNN-PSO exhibits a higher precision for the control of this nonlinear dynamical system. Control signal for test stage is illustrated in Fig. 5. According to these results, the MFLNN-PSO exhibits superior performance in comparison with the TRFN-G.

Example 2: In the second simulation example, a nonlinear process with longer input delays is intended to control. The difference equation of the time-delayed nonlinear dynamic system is [12]

$$\begin{aligned} &y_p(k+1) = 0{,}72y_p(k) + 0{,}025y_p(k-1)u(k-1) \\ &+0{,}01u^2(k-2) + 0{,}2u(k-3). \end{aligned} \quad (26)$$

In this nonlinear dynamic system, the actual output is a function of two past output signals and four past input signals. The desired signal used in train stage is

$$y_r(k+1) = \begin{cases} 10, if\ k \le 50\ or\ 100 < k \le 150 \\ 15, if\ 50 < k \le 100\ or\ 150 < k \le 200 \end{cases} \quad (27)$$

In addition, the same signal is utilized for the test phase. The control architecture in Fig. 3 is employed. The fitness value required for the train process is computed by

$$RMSE = \sqrt{\frac{1}{200}\sum_{k=1}^{200}(y_p(k+1) - (y_r(k+1))^2}. \quad (28)$$

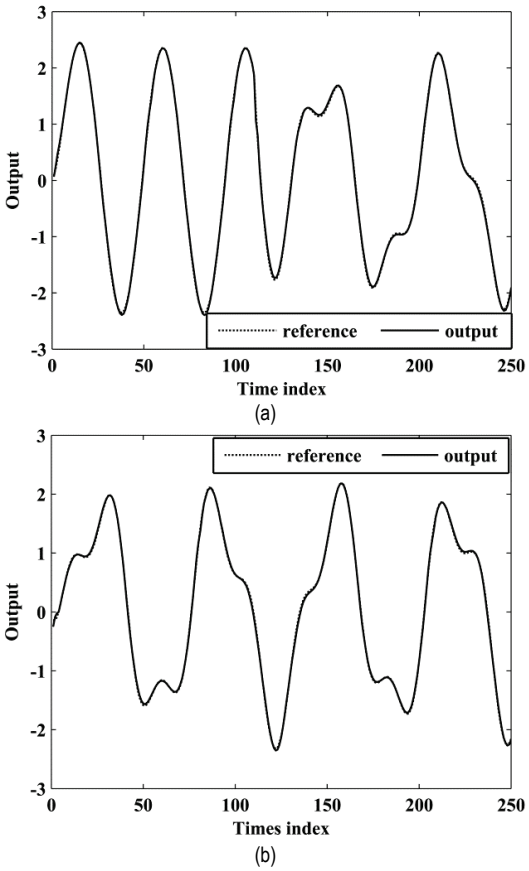


Figure 4 Desired and actual outputs for the example 1 (a) for train (b) for test

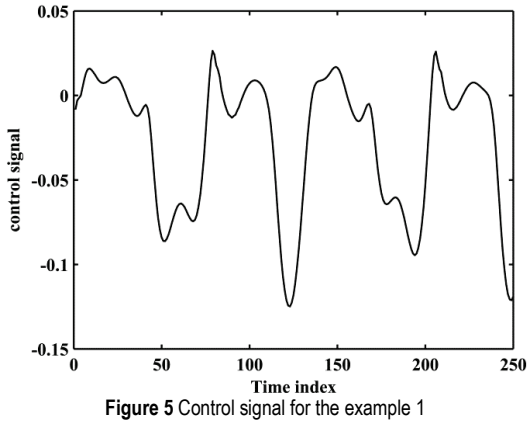


Figure 5 Control signal for the example 1

Once the train lasts 9000 time step, the MFLNN weights found by the PSO are saved. These weights values are utilized to assess the performance of the proposed MFLNN-PSO controller. In order to compare the MFLNN-PSO and the TRFN-G, simulation test results are presented in Tab. 4. Upon comparing the results, the proposed MFLNN-PSO controller in the

present work has obviously better performance than the TRFN-G controller proposed by [12]. As can be seen in Tab. 4, the proposed MFLNN-PSO accomplishes higher precision. The dynamic system outputs achieved after the train and desired signal are displayed in Fig. 6. Control signal for the test data is visualized in Fig. 7. It is easily seen in Fig. 7 that the proposed MFLNN-PSO controller expends less control efforts to follow the desired signal.

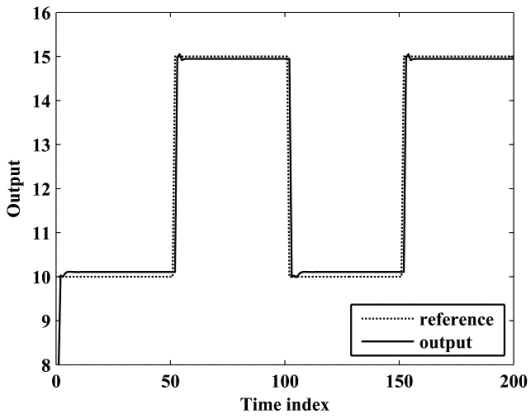


Figure 6 Desired and actual outputs for example 2

Table 4 RMSEs of the MFLNN-PSO and TRFN-G in Example 1

	TRFN-G	MFLNN-PSO	
RMSE_Train	1,1374	0,0087583	0,0082937
Parameter Size	48	42	62

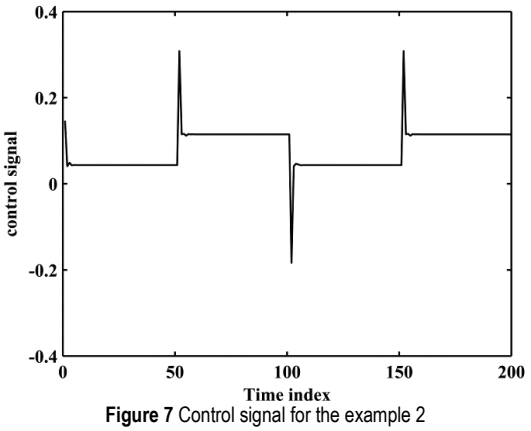


Figure 7 Control signal for the example 2

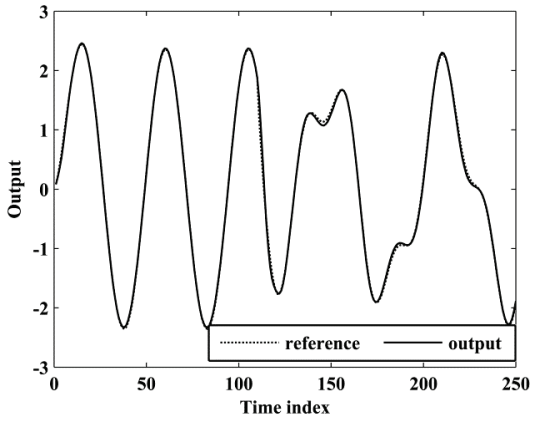
Example 3: In this simulation test, the desired signals employed for the train and test given by Eq. (23) and Eq. (24) which are the same as those in example 1 are utilized to control the HDD servo system given by Eq. (21). Like the previous examples, the neuro-control structure in Fig. 3 is utilized. 250 data are utilized for both test and train phases. The desired signals and outputs of the closed-loop system are pictured in Fig. 8. The output for the train signal is depicted in Fig. 8a, while that for the test is portrayed in Fig. 8b. The RMSE values from train and test results are given in Tab. 5. In the example, cost function is computed as follows:

$$RMSE = \sqrt{\frac{1}{250}\sum_{k=1}^{250}(y_p(k+1) - (y_r(k+1))^2}. \qquad (29)$$

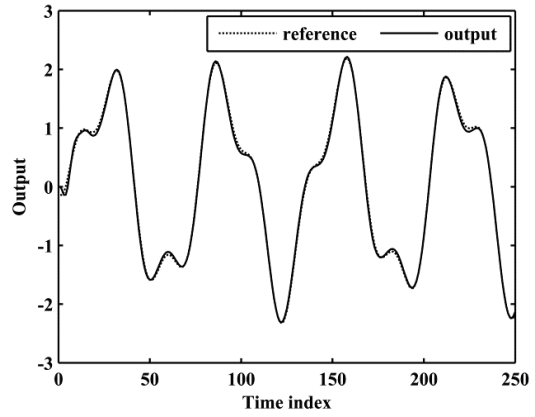
As seen in Fig. 8, the reference and process outputs are more or less equal. Hence, the proposed MFLNN-PSO controller tracks the reference closely. Control signal for the test data is depicted in Fig. 9.

Table 5 RMSEs of the MFLNN-PSO in Example 3

	MFLNN-PSO	
RMSE_Train	0,017298	0,014370
RMSE_Test	0,015376	0,012243
Parameter Size	42	62



(a)



(b)

Figure 8 Desired and actual outputs for example 3 (a) for train (b) for test

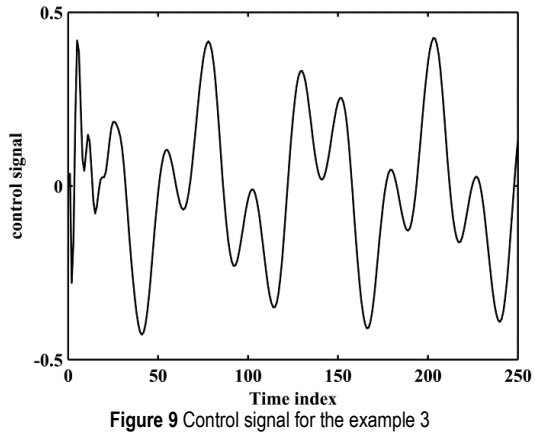


Figure 9 Control signal for the example 3

Example 4: In the last simulation example, in order to control the HDD servo system, the same desired signals as in example 2 are utilized. Both the train and test phases employ the same desired signals, which are given by Eq. (27). The HDD system equation given by Eq. (21) is employed to train and test the proposed MFLNN-PSO controller.

The convergent behaviours which are achieved from the train and test results are given in Tab. 6. Control

architecture like in aforementioned examples in Fig. 3 is utilized. The fitness value for PSO method to update the MFLNN weights is

$$\mathrm{RMSE} = \sqrt{\frac{1}{200}\sum_{k=1}^{200}(y_p(k+1) - (y_r(k+1))^2}\,. \qquad (30)$$

Table 6 RMSEs of the MFLNN-PSO in Example 4

	MFLNN-PSO	
RMSE_Train	0,079146	0,074763
Parameter Size	42	62

Following the train process, the desired signal and process output are shown in Fig. 10. Train process is kept on for 9000 time steps and the best weight values are employed in the test phase. Control signal for test data is illustrated in Fig. 11. The figure validates that the controller has a need of less control labours to follow the desired signal.

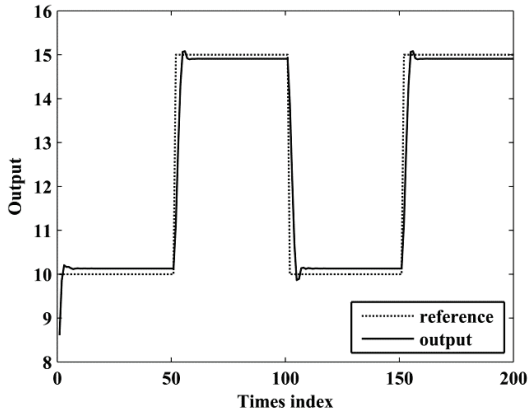


Figure 10 Desired and actual outputs for example 4

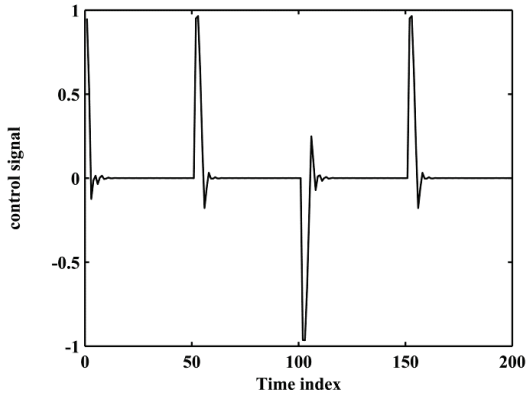


Figure 11 Control signal for example 4

5 CONCLUSION

In this study, a novel neuro-controller by using the Multifeedback-Layer Neural Network (MFLNN) and the particle swarm optimization (PSO) method is suggested to control the HDDservo systems in addition to nonlinear dynamical systems. The particle swarm optimization method is employed to update the weights of the Multifeedback-Layer Neural Network. Upon comparison of the proposed MFLNN-PSO controller with another control architecture referred to as the Takagi–Sugeno–Kang-type Recurrent Fuzzy Network with Genetic learning (TRFN-G) in the literature, the effectiveness and efficiency of the proposed MFLNN-PSO is proved. It is shown that the proposed MFLNN-PSO controller can be effectively implemented to the HDD servo systems besides the linear and nonlinear dynamic systems.

6 REFERENCES

[1] Beyhan, S. & Alci, M. (2010). Stable modeling based control methods using a new RBF network. *ISA transactions*, *49*(4), 510-518. https://doi.org/10.1016/j.isatra.2010.04.005

[2] Kim, Y. H. & Lewis, F. L. (2000). Optimal design of CMAC neural-network controller for robot manipulators. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *30*(1), 22-31. https://doi.org/10.1109/5326.827451

[3] Lewis, F. L., Yesildirek, A., & Liu, K. (1996). Multilayer neural-net robot controller with guaranteed tracking performance. *IEEE Transactions on Neural Networks*, *7*(2), 388-399. https://doi.org/10.1109/72.485674

[4] San, P. P., Ren, B., Ge, S. S., Lee, T. H., & Liu, J. K. (2011). Adaptive neural network control of hard disk drives with hysteresis friction nonlinearity. *IEEE Transactions on Control Systems Technology*, *19*(2), 351-358. https://doi.org/10.1109/TCST.2010.2041233

[5] Lee, C. H. & Teng, C. C. (2000). Identification and control of dynamic systems using recurrent fuzzy neural networks. *IEEE Transactions on fuzzy systems*, *8*(4), 349-366. https://doi.org/10.1109/91.868943

[6] Lin, F. J., Lin, C. H., & Huang, P. K. (2004). Recurrent fuzzy neural network controller design using sliding-mode control for linear synchronous motor drive. *IEE Proceedings-Control Theory and Applications*, *151*(4), 407-416. https://doi.org/10.1049/ip-cta:20040652

[7] Lin, F. J. & Wai, R. J. (2003). Robust recurrent fuzzy neural network control for linear synchronous motor drive system. *Neurocomputing*, *50*, 365-390. https://doi.org/10.1016/S0925-2312(02)00572-6

[8] Mastorocostas, P. A. & Theocharis, J. B. (2002). A recurrent fuzzy-neural model for dynamic system identification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *32*(2), 176-190. https://doi.org/10.1109/3477.990874

[9] Chow, T. W. & Fang, Y. (1998). A recurrent neural-network-based real-time learning control strategy applying to nonlinear systems with unknown dynamics. *IEEE transactions on industrial electronics*, *45*(1), 151-161. https://doi.org/10.1109/41.661316

[10] Coban, R. (2013). A context layered locally recurrent neural network for dynamic system identification. *Engineering Applications of Artificial Intelligence*, *26*(1), 241-250. https://doi.org/10.1016/j.engappai.2012.09.023

[11] Kim, Y. H., Lewis, F. L., & Abdallah, C. T. (1997). A dynamic recurrent neural-network-based adaptive observer for a class of nonlinear systems. *Automatica*, *33*(8), 1539-1543. https://doi.org/10.1016/S0005-1098(97)00065-4

[12] Juang, C. F. (2002). A TSK-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithms. *IEEE Transactions on Fuzzy Systems*, *10*(2), 155-170. https://doi.org/10.1109/91.995118

[13] Shi, Y. & Eberhart, R. C. (1999). Empirical study of particle swarm optimization. In *Evolutionary computation, 1999. CEC 99. Proceedings of the 1999 congress on* (Vol. 3, pp. 1945-1950). IEEE. https://doi.org/10.1109/CEC.1999.785511

[14] Dong, X., Zhao, Y., Xu, Y., Zhang, Z., & Shi, P. (2011). Design of PSO fuzzy neural network control for ball and plate system. *International Journal of Innovative Computing, Information and Control*, *7*(12), 7091-7103.

[15] Sheikhan, M., Hemmati, E., & Shahnazi, R. (2017). GA-PSO-optimized neural-based control scheme for adaptive congestion control to improve performance in multimedia applications. *arXiv preprint arXiv:1711.06317*.

[16] Tsekouras, G. E. (2013). A simple and effective algorithm for implementing particle swarm optimization in RBF network's design using input-output fuzzy clustering. *Neurocomputing*, *108*, 36-44. https://doi.org/10.1016/j.neucom.2012.11.011

[17] Junyou, B. (2007, September). Stock Price forecasting using PSO-trained neural networks. In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on* (pp. 2879-2885). IEEE. https://doi.org/10.1109/CEC.2007.4424837

[18] Mendes, R., Cortez, P., Rocha, M., & Neves, J. (2002). Particle swarms for feedforward neural network training. In *Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on* (Vol. 2, pp. 1895-1899). IEEE. https://doi.org/10.1109/IJCNN.2002.1007808

[19] Savran, A. (2007). Multifeedback-layer neural network. *IEEE Transactions on Neural Networks*, *18*(2), 373-384. https://doi.org/10.1109/TNN.2006.885439

[20] Aksu, I. O. & Coban, R. (2013, May). Identification of disk drive systems using the multifeedback-layer neural network and the particle swarm optimization algorithm. In *Technological Advances in Electrical, Electronics and Computer Engineering (TAEECE), 2013 International Conference on* (pp. 230-234). IEEE. https://doi.org/10.1109/TAEECE.2013.6557276

[21] Ren, X., Lewis, F. L., Zhang, J., & Ge, S. S. (2009). Feedforward control based on neural networks for hard disk drives. *IEEE Transactions on Magnetics*, *45*(7), 3025-3030. https://doi.org/10.1109/TMAG.2009.2015660

[22] Atsumi, T. & Messner, W. C. (2013). Estimation method for unobservable settling vibration of head-positioning control in hard disk drives. *Mechatronics*, *23*(1), 37-45. https://doi.org/10.1016/j.mechatronics.2012.10.007

[23] Chen, B. M., Lee, T. H., Peng, K., & Venkataramanan, V. (2006). Hard Disk Drive Servo Systems (2nd edn). Advances in Industrial Control.

[24] Pérez-Arancibia, N. O., Tsao, T. C., & Gibson, J. S. (2010). A new method for synthesizing multiple-period adaptive–repetitive controllers and its application to the control of hard disk drives. *Automatica*, *46*(7), 1186-1195. https://doi.org/10.1016/j.automatica.2010.04.007

[25] Aksu, I. O. & Coban, R. (2013, November). Training the multifeedback-layer neural network using the Particle Swarm Optimization algorithm. In *Electronics, Computer and Computation (ICECCO), 2013 International Conference on* (pp. 172-175). IEEE. https://doi.org/10.1109/ICECCO.2013.6718256

[26] Kennedy, J. & Eberhart, R. (1995). PSO optimization. In *Proc. IEEE Int. Conf. Neural Networks* (Vol. 4, pp. 1941-1948). IEEE Service Center, Piscataway, NJ.

[27] Kelemen, A., Abraham, A., & Chen, Y. (Eds.). (2008). *Computational intelligence in bioinformatics* (Vol. 94). Springer.

[28] Franklin, G. F., Powell, J. D., & Workman, M. L. (1998). *Digital control of dynamic systems* (Vol. 3). Menlo Park, CA: Addison-wesley.

[29] Hughes, G. F. (2002). Wise drives [hard disk drive]. *IEEE Spectrum*, *39*(8), 37-41. https://doi.org/10.1109/MSPEC.2002.1021942

[30] Dorf, R. C., & Bishop, R. H. (2011). *Modern control systems*. Pearson.

[31] Rahman, M. A., Al Mamun, A., & Yao, K. (2015). Discrete time adaptive controller for suppression of resonance in hard disk drive servo system. *International Journal of Control, Automation and Systems*, *13*(5), 1161-1172. https://doi.org/10.1007/s12555-014-0231-0

[32] Rahman, M. A., Mamun, A. A., & Yao, K. (2016). Suppression of resonance in hard disk drive servo system with feedback resonant controller: a benchmark model study. *International Journal of Modelling, Identification and Control*, *26*(2), 171-184. https://doi.org/10.1504/IJMIC.2016.078325

Contact information:

Ramazan COBAN, Assoc. Prof. Dr.
Department of Computer Engineering,
Cukurova University
01330 Balcali, Saricam / Adana / Turkey
E-mail: rcoban@cu.edu.tr

Inayet Ozge AKSU, Res. Asst.
Department of Computer Engineering,
Adana Science and Technology University,
Gültepe Mahallesi, Çatalan Caddesi No:201/5
01250 Sarıçam / Adana / Turkey
E-mail: oaksu@adanabtu.edu.tr