Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2013) 21: 793 – 803

doi:10.3906/elk-1109-58

Controlling the chaotic discrete-Hénon system using a feedforward neural network with an adaptive learning rate

Kürşad GÖKCE,[1,*] **Yılmaz UYAROĞLU**[2]

[1]Department of Electrical and Electronics Engineering, Institute of Science and Technology, Sakarya University, Sakarya, Turkey

[2]Department of Electrical and Electronics Engineering, Engineering Faculty, Sakarya University, Sakarya, Turkey

Received: 28.01.2011 • **Accepted:** 23.02.2012 • **Published Online:** 03.05.2013 • **Printed:** 27.05.2013

Abstract: This paper proposes a feedforward neural network-based control scheme to control the chaotic trajectories of a discrete-Hénon map in order to stay within an acceptable distance from the stable fixed point. An adaptive learning back propagation algorithm with online training is employed to improve the effectiveness of the proposed method. The simulation study carried in the discrete-Hénon system verifies the validity of the proposed control system.

Key words: Neural network, backpropagation, online training, chaotic systems, adaptive learning, Hénon map

1. Introduction

Controlling chaotic systems via different control schemes is an increasingly important field of research and also has practical importance since chaotic systems are involved more and more in the fields of engineering, chemistry, physics, biology, and mathematics.

Notably, neural network-based control algorithms are widely used for systems with chaotic behaviors. The primary advantage of these algorithms over other control algorithms is that they utilize only the system inputs and can be used in uncertain nonlinear systems with noise. These algorithms are also used in cryptology. In [1], an artificial neural network-based chaotic generator was proposed for encryption to overcome the weakness of chaotic systems such as synchronization and fewness of parameters.

Chaotic systems are sensitive to initial conditions and it is difficult to determine their future behavior. Some researchers have studied the nonlinear behavior of such systems. In [2], the theory of chaotic systems was reviewed and the features of the high dimensional attractors were presented. In [3], Pehlivan and Uyaroğlu introduced a new 3-dimensional quadratic autonomous chaotic system that exhibits Lorenz-like attractors and they performed simulations and an experimental study of the chaotic system. Another study was performed on the implementation of the chaotic system with nonlinear function blocks by field programmable analogue array programming in order to model the chaotic system [4]. The control problem of chaotic systems has also been debated by researchers using different approaches. In [5], the authors proposed to stabilize the unstable periodic orbits by making small perturbations on some available parameters of the chaotic system. In [6], Grebogi aimed to select a target chaotic orbit corresponding to a desirable system performance and applied a feedback control to stabilize a trajectory with a random initial condition around this target orbit. A similar study was examined in [7] to stabilize the unstable periodic orbits using a time-delayed control method based on the optimal control

*Correspondence: kursad1258@gmail.com

principle. The work presented in [8] showed that 2 chaotic signals can be controlled simultaneously using a back propagation neural network. In [9], a method for the simultaneous control of deterministic chaos in the Lozi, Ikeda, and Tinkerbell systems, by utilizing a radial basis function network, was presented. In [10], Yang et al. proposed a variable structure control approach to stabilize a chaotic Hénon map and to synchronize 2 Hénon chaotic systems.

Most of the research was performed for understanding and controlling chaotic behaviors, which are irregular and sensitive to the initial conditions. On the other hand, there is some research intended to improve the control algorithms used in chaotic systems. One of these was given in [11]. The main idea of this study was to accelerate the conventional back propagation algorithm by making weight extrapolations. The authors in [12] developed a harmony search algorithm that does not require initial values and uses a random search instead of a gradient search. They applied this algorithm to improve the optimization problem in chaotic systems and to synchronize the 2 Hénon chaotic systems. An optimal control technique to control the trajectories of the Hénon map by means of a support vector machine controller with a radial basis function kernel was proposed in [13].

In this paper, we propose a simple solution to control problems in chaotic systems and reduce the computational effort using an improved back propagation algorithm with an adaptive learning rate. As an example, this control approach has been applied to the chaotic discrete Hénon system to control its trajectories within an $\in$-neighborhood of the fixed point, even if there are some bounded noise parameters in the system. Expectedly, the simulation results show the effectiveness of the proposed method.

2. Previous studies

There are some studies in the literature that employ neural networks to control the chaotic trajectories in discrete chaotic systems. A back propagation neural network has been studied and tested on the Hénon and logistic maps in [14]. However, this study was concerned only with the periodic motion of the chaotic systems and did not guarantee the stability of the chaos on the fixed point, as shown in Figure 1. In another study [15], the proposed scheme in [5] was applied on the Ikeda map in order to stabilize the unstable Period 1 orbit of the Ikeda map, as shown in Figure 2.

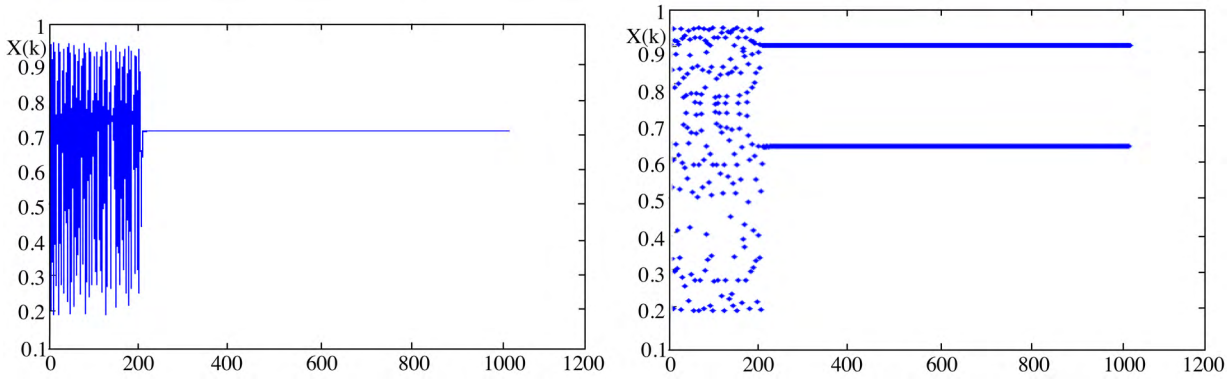

Figure 1. Period 1 and Period 2 trajectory [14].

In [16], a back propagation neural network has been studied and tested on the Ikeda map in order to make stable unsteady periodic orbits. This paper employs the classical back propagation neural network algorithm

and investigates how the number of hidden units and the different sets of input patterns affect the performance of the network.

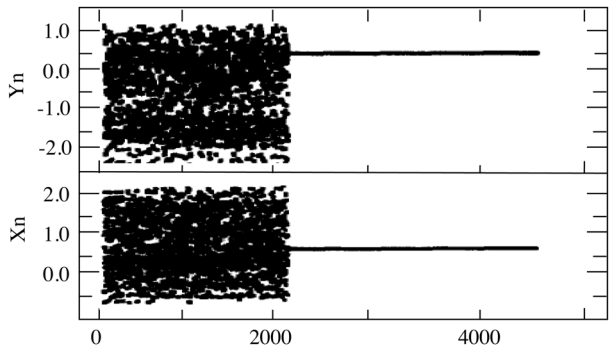

Figure 2. Period 1 orbit of the Ikeda map [15].

This study differs from the previous studies in 2 ways. First, an adaptive learning rate is used in the neural network training algorithm in order to improve the stability. Second, we focused not only on the stabilization of the specific periodic orbits in the chaotic systems, but also on the stabilization of the complete chaotic trajectory on the fixed point. Furthermore, it is worth noting that in previous chaotic control papers, the effects of the epsilon neighborhood of the fixed point have not been taken into account in detail. Hence, with respect to the previous works, the other feature of this paper is that it investigates how the chosen parameter epsilon has impacted on the stability of the chaotic trajectory.

3. Problem statement

A simple 2-dimensional evolution map found by French astronomer Michael Hénon [17] is given by:

$$\varphi(x_t, y_t) = (x_{t+1}, y_{t+1}) = (1 - ax_t^2 + y_t, bx_t), \tag{1}$$

where $\varphi : R^2 \to R^2$, a, and b are 2 real parameters and are chosen as a $= 1.4$ and b $= 0.3$ to study the Hénon system. With these parameters, Eq. (1) has the fundamental characteristics of a chaotic system, which means it is ergodic, dissipative, and sensitive to the initial conditions. A Hénon map is also a strange attractor with period doubling bifurcation. One can find a detailed bifurcation analysis of a Hénon map in [18]. A Hénon map has 2 fixed points:

$$x_{1,2} = \frac{-(1-b) \pm \sqrt{(1-b)^2 + 4a}}{2a}, y_{1,2} = bx_{1,2}. \tag{2}$$

For $\frac{(1-b)^2}{4} < a < \frac{3(1-b)^2}{4}$, the attractor is in the stable leaf of a point. If one tries to exceed this interval slightly, the attractor behaves as a strange attractor.

The objective of this study is designing a feedforward neural network-based control scheme with an adaptive learning rate such that the iterated trajectory of Eq. (1) falls into the $\in$-neighborhood of the fixed point, even if there are some bounded noise parameters added to the input patterns.

4. Proposed neural network control scheme

The proposed neural network structure with online training for the 2-dimensional discrete Hénon system in Eq. (1) can be illustrated as in Figure 3.

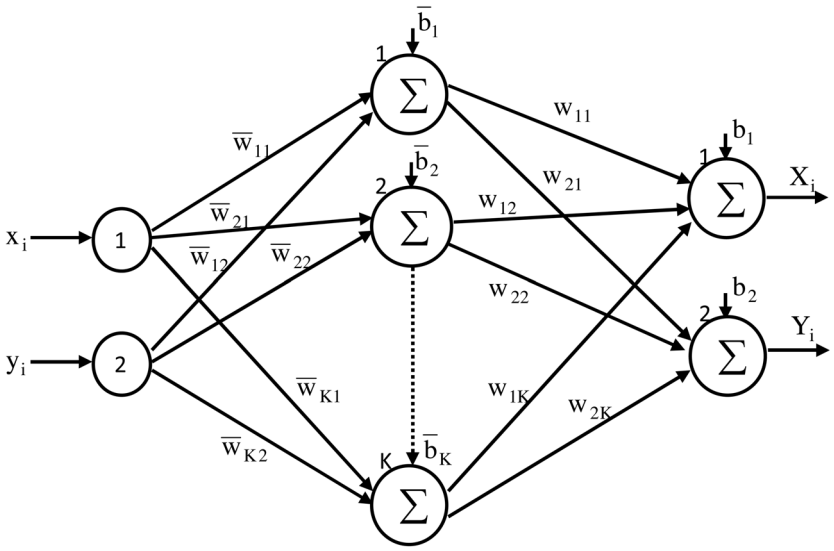

Figure 3. Proposed neural network structure.

Here, $(x_i, y_i)i = 1, 2, ...N$ are the input patterns with bounded noise parameters that are generated iteratively from Eq. (1) during the online training. N is the number of total input pairs and is also equal to the number of iterations. (X_i, Y_i) are the output patterns that should have to be approximated to the fixed points, (T_x, T_y), as much as possible.

One hidden layer with $k = 1, 2.....K$ neurons is used for training. The input to kth unit of the hidden layer is denoted by I_k and is given by:

$$I_k = x_i\overline{w_{k1}} + y_i\overline{w_{k2}} + \overline{b_k}, i = 1, 2, ...N, k = 1, 2, ...K,$$

where $\overline{w_{k1}}$ and $\overline{w_{k2}}$ are the weights between the kth neuron of the hidden layer and the input data x_i and y_i, respectively. $\overline{b_k}$ is the bias value of the kth neuron in the hidden layer. By applying the tangent sigmoid activation function F on I_k, the output $\overline{O_{pk}}$ is obtained as follows:

$$F(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}, \tag{3}$$

$$\overline{O_{pk}} = F(x_i\overline{w_{k1}} + y_i\overline{w_{k1}} + \overline{b_k}). \tag{4}$$

The network outputs are given by:

$$X_i = F\left(\sum_{k=1}^{K} \overline{O_{pk}} w_{1k} + b_1\right), \tag{5}$$

$$Y_i = F\left(\sum_{k=1}^{K} \overline{O_{pk}} w_{2k} + b_2\right). \tag{6}$$

From the network outputs in Eqs. (5) and (6), and the fixed points, the error functions E_{x_i}, E_{y_i}, which will be minimized by adjusting the weights and bias values, are given by:

$$E_{x_i} = T_x - X_i, \tag{7}$$

$$E_{y_i} = T_y - Y_i. \tag{8}$$

4.1. Derivation of the updating rules of the weights and bias values

In this section, we present the updating rules of the weights and bias values of the proposed neural network by utilizing the popular conjugate gradient method. The following updating equations for the weights and bias values between the output and hidden layers are given by:

$$\Delta w_{1k}(t+1) = \eta_x \delta_{1k} \overline{O_{pk}}, t = 1, 2, ...N, \tag{9}$$

$$\Delta w_{2k}(t+1) = \eta_y . \delta_{2k} . \overline{O_{pk}}, \tag{10}$$

$$\delta_{1k} = E_{x_i} . X_i . (1 - X_i), \tag{11}$$

$$\delta_{2k} = E_{y_i} . Y_i . (1 - Y_i), \tag{12}$$

$$\Delta b_1(t+1) = \eta_x . \delta_{1k}, \tag{13}$$

$$\Delta b_2(t+1) = \eta_y . \delta_{2k}, \tag{14}$$

$$w_{1k_{new}} = w_{1k_{old}} + \Delta w_{1k}(t+1), \tag{15}$$

$$w_{2k_{new}} = w_{2k_{old}} + \Delta w_{2k}(t+1), \tag{16}$$

$$b_{1_{new}} = b_{1_{old}} + \Delta b_1(t+1), \tag{17}$$

$$b_{2_{new}} = b_{2_{old}} + \Delta b_2(t+1) \quad , \tag{18}$$

where η_x and η_y are the adaptive learning rates. In [19] a learning rate, β, is asserted, which is varied according to the error function such that if the error is less than the previous value, then β is multiplied by a factor, $\varphi > 1$ or if the error is more than a few percent above the previous value, then β is multiplied by a factor $\varsigma < 1$ for the next iteration at the end of the updating procedure. This rule is applied until a desired error response, E_{th}, is obtained. Hence, η_x and η_y can be updated as:

$$\eta_x = \begin{cases} \eta_x . \varphi, if \left| E_{x_{i+1}} \right| < |E_{x_i}| \\ \eta_x . \varsigma, \text{ otherwise} \\ \\ \eta_x, |E_{x_i}| \leq E_{th} \text{ for } \forall i \in N \end{cases} , \tag{19}$$

$$\eta_y = \begin{cases} \eta_y . \varphi, if \left| E_{y_{i+1}} \right| < |E_{y_i}| \\ \eta_y . \varsigma, \text{ otherwise} \\ \eta_y, |E_{y_i}| \leq E_{th} \text{ for } \forall i \in N \end{cases} . \tag{20}$$

In a similar way, the following updating equations for the weights and the bias values between the hidden layer and the input layer are obtained as:

$$\Delta \overline{w_{k1}}(t+1) = \eta_z \overline{\delta_{k1}} x_i \quad , \tag{21}$$

$$\Delta \overline{w_{k2}}(t+1) = \eta_z \overline{\delta_{k1}} y_i \quad , \tag{22}$$

$$\overline{\delta_{k1}} = (\delta_{1k} w_{1k_{old}} + \delta_{2k} w_{2k_{old}}) \overline{O_{pk}} (1 - \overline{O_{pk}}), \tag{23}$$

$$\Delta \overline{b_k}(t+1) = \eta_z . \overline{\delta_{k1}},$$

$$\overline{w_{k1_{new}}} = \overline{w_{k1_{old}}} + \Delta\overline{w_{k1}}(t+1), \tag{24}$$

$$\overline{w_{k2_{new}}} = \overline{w_{k2_{old}}} + \Delta\overline{w_{k2}}(t+1), \tag{25}$$

$$b_{k_{new}} = b_{k_{old}} + \Delta\overline{b_k}(t+1), \tag{26}$$

where η_z is also changed adaptively as follows:

$$\eta_z = \begin{cases} \eta_z.\varphi, if \left|E_{x_{i+1}}\right| + \left|E_{y_{i+1}}\right| < |E_{x_i}| + |E_{y_i}| \\ \eta_z.\varsigma, \text{ otherwise} \\ \eta_z, |E_{x_i}| + |E_{y_i}| \leq E_{th} \text{ for } \forall i \in N \end{cases} . \tag{27}$$

All of these updating rules satisfy the desired response such that the trapping region of the system in Eq. (1) falls into the $\in$-neighborhood of the stable fixed point.

5. Simulation study

In the simulation, we proposed the following learning procedure for the system, Eq. (1).

Input layer: Input patterns are generated by the system, Eq. (1), with the initial point $(x_o, y_o) \rightarrow (0.1, 0.2)$. Each input pair is added with the noise parameter, which is distributed at the interval of (–0.01, 0.01), and presented to the network for online training. The number of input patterns is 1500 for this simulation.

Hidden layer: We use 1 hidden layer with 10 neurons to achieve the best performance.

Output layer: The output patterns are generated by the neural network and must be controlled around the fixed points. For the system, Eq. (1), with a = 1.4 and b = 0.3, there are 2 fixed points as follows:

$$\begin{array}{l} (T_{x1}, T_{y1}) \approx (0.63, 0.19) \\ (T_{x2}, T_{y2}) \approx (-1.13, -0.34) \end{array} .$$

Here, we consider only the first one because this point lies inside the trapping region of the system, Eq. (1).

When the learning is finished for these input patterns, which are obtained for the initial point (0.1, 0.2), other input patterns are generated by starting with another initial point and each of them are mixed with randomly changing noise parameters and then they are presented to the neural network. This learning stage continues iteratively until achieving the desired error response. In our simulation $\in$ is selected as a small number of 0.005.

Under these simulation conditions, the following results are quite satisfactory. Figure 4 shows the trajectories of the 2-dimensional discrete Hénon map with the initial point (0.1, 0.2) for 1500 patterns before training.

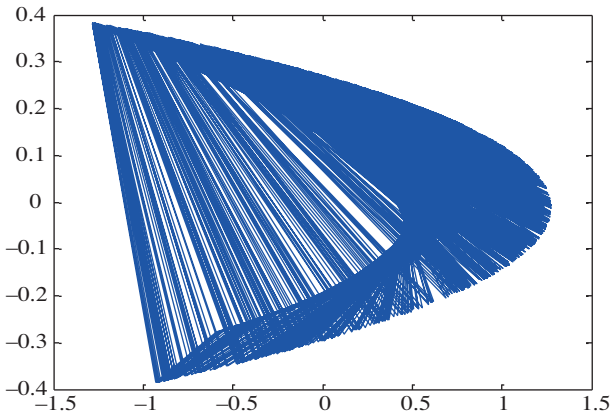

Figure 4. Trajectories of the system, Eq. (1), with the initial point (0.1, 0.2).

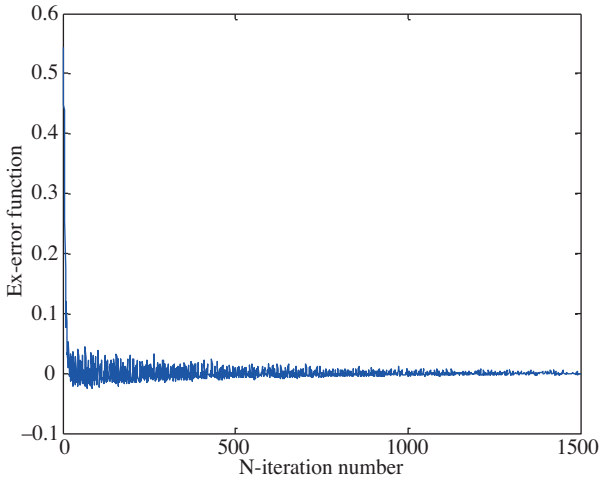

Figure 5. The variation of error function E_{x_i} during the training period.

In order to check the performance of the trained network, we present a test set that includes 500 input patterns that are generated by the system, Eq. (1), for 3 different initial points, (0, 0), (0.6, 0.3), and (0.5, 0.5). Randomly distributed noise parameters at the interval of (–0.01, 0.01) are also added to the input patterns. The effectiveness of our method can be seen in the following figures. Figure 7 shows the trajectory of the tested input patterns with the initial point (0, 0) and he $\in$-neighborhood of the fixed point (0.63, 0.19) in green. Figure 8 shows how the trajectory seen in Figure 7 falls into this green circle after applying the proposed neural network control method. Similarly, Figures 9 and 10 show the trajectory for the initial point (0.6, 0.3) and Figures 11 and 12 show the trajectory for the initial point (0.5, 0.5).

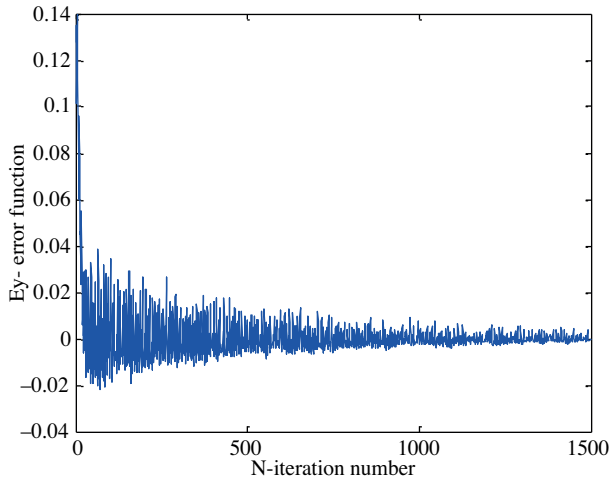

Figure 6. The variation of error function E_{y_i} during the training period.

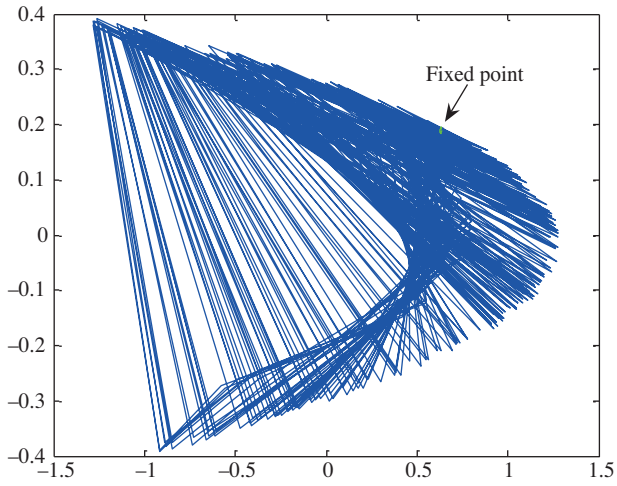

Figure 7. Trajectories of the tested input patterns with the initial point (0, 0).

The results of Figures 7 and 8 are about the same as the results of Figures 9, 10, 11, and 12. We have obtained these results for 3 different initial points.

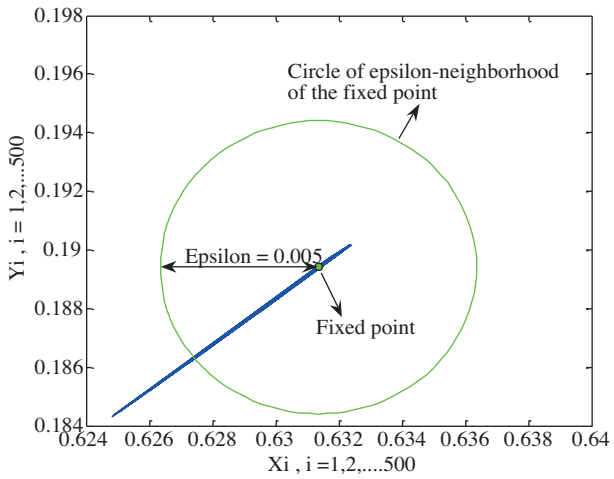

Figure 8. Trajectories of the output patterns of the neural network for Figure 7.

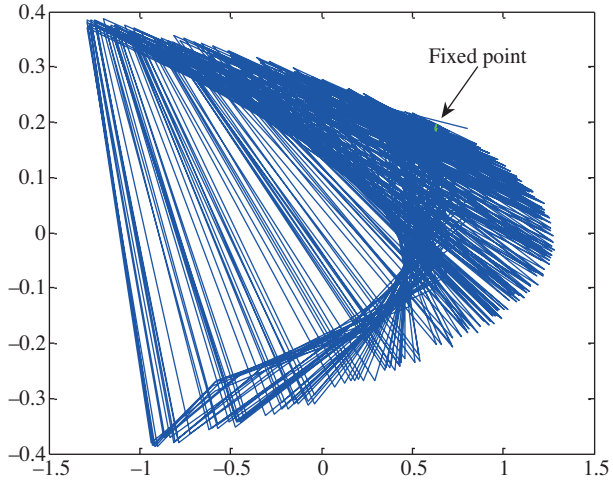

Figure 9. Trajectories of the tested input patterns with the initial point (0.6, 0.3).

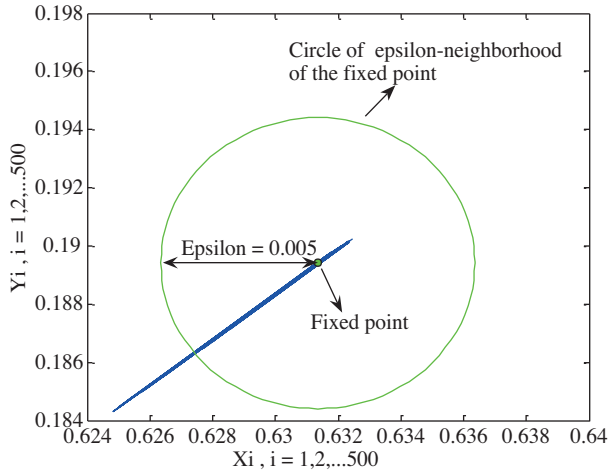

Figure 10. Trajectories of the output patterns of the neural network for Figure 9.

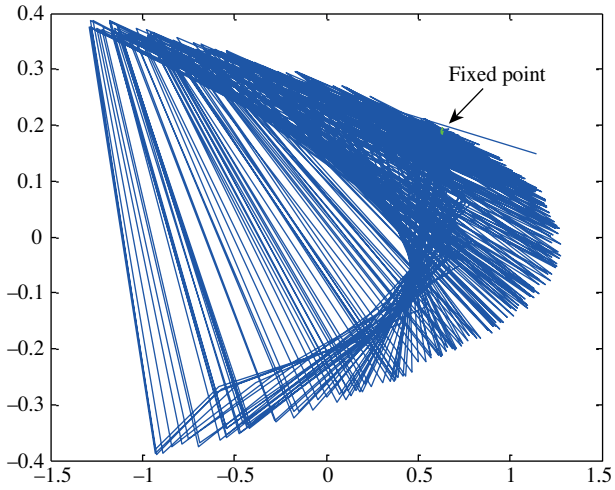

Figure 11. Trajectories of the tested input patterns with the initial point (0.5, 0.5).

Table 1 also shows a performance comparison of the proposed method for different $\in$ values and initial points.

Table 1. The performance comparison of the proposed method for different $\in$ values.

	Starting (initial) points		
eps	(0, 0)	(0.6, 0.3)	(0.5, 0.5)
0.001	351	352	361
0.002	402	397	402
0.003	429	429	439
0.004	448	448	457
0.005	472	470	476

Each trajectory consists of 500 points. We used the parameter $\in$= 0.005 in our simulation. The numbers

in Table 1 indicate how many points in the trajectory fall into the $\in$-neighborhood of the fixed point (0.63, 0.19). As seen from Table 1, if one tries to increase the $\in$ value step by step, the number of the points inside the $\in$-neighborhood of the fixed point will increase. $\in$ depends on the desired sensitivity chosen by the applicator and varies in the application.

The effectiveness of the adaptive learning rate in the training algorithm has also been demonstrated in our simulation. Figures 13 and 14 show the trajectories of the output patterns of the neural network for the adaptive and fixed learning rates, respectively, under the same initial condition (0, 0) and the same $\in$ value 0.005. In the simulation, the fixed learning rates were chosen as $\eta_x = \eta_y = \eta_z = 0.1$.

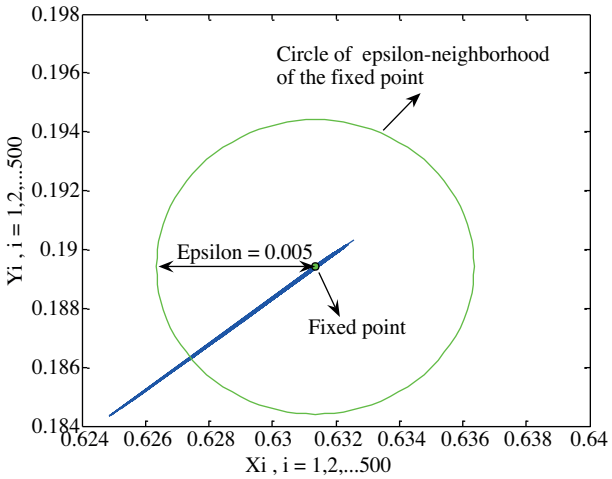

Figure 12. Trajectories of the output patterns of the neural network for Figure 11.

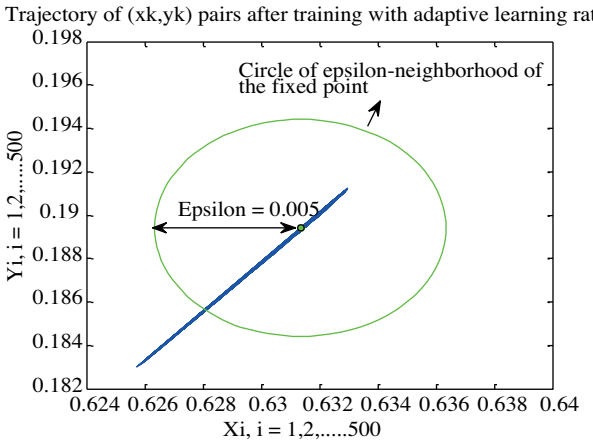

Figure 13. Trajectories of the output patterns of the neural network with the adaptive learning rates.

Figure 15 also shows the error trajectories during the training period for each case (i.e. the fixed and adaptive learning rates).

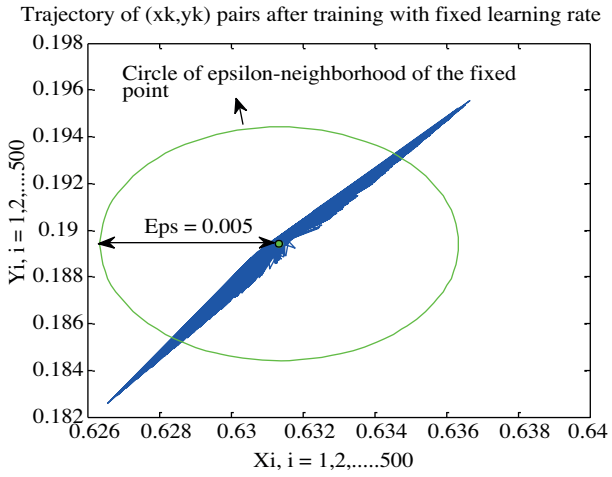

Figure 14. Trajectories of the output patterns of the neural network with the fixed learning rates ($\eta_x = \eta_y = \eta_z = 0.1$).

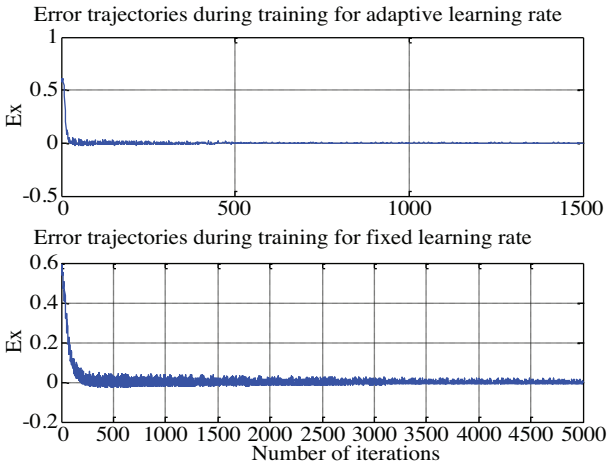

Figure 15. Error trajectories during the training period for the fixed (below) and adaptive learning rates (above).

As seen from Figure 15, it is very obvious that the learning times are quite long when obtaining almost the same performance when using the fixed learning rates. The number of iterations is 5000 for the fixed learning rates; however, it is 1500 for the adaptive learning rates. Table 2 summarizes the results of the performance of the proposed method in the case of using the fixed and the adaptive learning rates.

Table 2. The performance comparison of the proposed method in the case of using the fixed and the adaptive learning rates.

	For the same initial (0, 0) point, the same number of input patterns (500), and the same $\in$ value (0.005)	
	Number of iterations during training	Number of points in the $\in -$neighborhood of the fixed point after training
Fixed learning rates	5000	415
Adaptive learning rates	1500	454

6. Conclusions

In this paper, a back propagation algorithm with an adaptive learning rate has been developed to control the chaotic trajectories of the 2-dimensional discrete Hénon map in the stable fixed point. The performance comparisons of the proposed method show that an adaptive learning rate can significantly increase the performance of the neural network and decrease the learning times during training. We have also investigated the effect of the different initial points and the design parameter $\in$ on the stability of the chaotic system. The results have also demonstrated that the effectiveness of our method is valid, even if the input pattern of the chaotic system is noisy. The updating rules developed in this study can also be applied to other 2-dimensional chaotic systems and be easily extended to multidimensional ones.

References

[1] İ. Dalkıran, K. Danışman, "Artificial neural network based chaotic generator for cryptology", Turkish Journal of Electrical Engineering & Computer Sciences, Vol. 18, pp. 225–240, 2010.

[2] L. Shilnikov, "Mathematical problems of nonlinear dynamics: a tutorial", Journal of the Franklin Institute, Vol. 334B, pp. 793–864, 1997.

[3] İ. Pehlivan, Y. Uyaroğlu, "A new chaotic attractor from general Lorenz system family and its electronic experimental implementation", Turkish Journal of Electrical & Computer Sciences, Vol. 18, pp. 171–184, 2010.

[4] R. Kılıç, F.Y. Dalkıran, "Programmable design and implementation of a chaotic system utilizing multiple nonlinear functions", Turkish Journal of Electrical & Computer Sciences, Vol. 18, pp. 647–656, 2010.

[5] E. Ott, C. Grebogi, J.A. Yorke, "Controlling chaos", Physical Review Letters, Vol. 64, pp. 1196–1199, 1990.

[6] C. Grebogi, "Control and applications of chaos", Journal of the Franklin Institute, Vol. 334B, pp. 1115–1146, 1997.

[7] Y.P. Tian, X. Yu, "Stabilizing unstable periodic orbits of chaotic systems via an optimal principle", Journal of the Franklin Institute, Vol. 337, pp. 771–779, 2000.

[8] C. Hernández, J. Castellanos, R. Gonzalo, V. Palencia, "Neural control of chaos and applications", International Journal on Information Theory and Applications, Vol. 12, pp. 103–110, 2008.

[9] A. Castellanos, R. Gonzalo, A. Martinez, "Simultaneous control of chaotic systems using RBF networks", 6th International Conference on Information Research and Applications, 2008.

[10] Z. Yang, Q. Yao, C. Yang,, "Control and synchronization of Hénon chaos via a novel variable structure control", Dynamics of Continuous, Discrete and Impulsive Systems Series B, Vol. 11, pp. 665–672, 2004.

[11] S.V. Kamarthi, S. Pittner, "Accelerating neural network training using weight extrapolations", Neural Networks, Vol. 12, pp. 1285–1299, 1999.

[12] L. dos Santos Coelho, D.L. de Andrade Bernert, "An improved harmony search algorithm for synchronization of discrete-time chaotic systems", Chaos, Solitons and Fractals, Vol. 41, pp. 2526–2532, 2009.

[13] J.A.K. Suykens, J. Vandewalle, "Chaos control using least-squares support vector machines", International Journal of Circuit Theory and Applications, Vol. 27, pp. 605–615, 1999.

[14] X.P. Zong, Y. Geng, "Control chaotic systems based on BP neural network with a new perturbation", International Conference on Wavelet Analysis and Pattern Recognition, pp. 166–170, 2009.

[15] M.P. Alsing, R. Gavrielides, V. Kovanis, "Controlling unstable periodic orbits in a nonlinear optical system: the Ikeda map", Nonlinear Optics: Materials, Fundamentals and Applications, pp. 72–74, 1994.

[16] C. Hernandez, A. Martinez, J. Castellanos, F.L Mingo, "Controlling chaotic nonlinear dynamical systems", Proceedings of the 6th IEEE International Conference on Electronics, Circuits and Systems,, Vol. 3, pp. 1231–1234, 1999.

[17] M. Hénon, "A two-dimensional mapping with a strange attractor", Communications in Mathematical Physics, Vol. 50, pp. 69–77, 1976.

[18] C. Murakami, W. Murakami, K. Hirose, "Sequence of global period doubling bifurcation in the Hénon maps", Chaos, Solitons and Fractals, Vol. 14, pp. 1–17, 2002.

[19] L.M. Saini, "Peak load forecasting using Bayesian regularization resilient and adaptive back propagation learning based artificial neural networks", Electric Power Systems Research, Vol. 78, pp. 1302–1310, 2008.