Analysis of the dynamical behavior of a feedback auto-associative memory

Mahmood Amiri, Sohrab Saeb*, Mohammad Javad Yazdanpanah, S. Ali Seyyedsalehi

Faculty of Biomedical Engineering, Amirkabir University of Technology (Tehran Polytechnic), 424 Hafez Ave, Tehran 15875-4413, Iran

Abstract

The dynamical behavior and the stability properties of fixed points in a feedback auto-associative memory are investigated. The proposed structure encompasses a multi-layer perceptron (MLP) and a feedback connection that links input and output layers through delay elements. The MLP is initially trained so that it maps the training patterns into themselves as an auto-associative memory. The feedback connection is then established in order to make the feedback auto-associative memory. We derive some explicit equations based on the theory of dynamical systems, which relate the stability properties of fixed points to the network parameter values. We then perform some case studies for the purpose of performance comparisons between the proposed model and a self-feedback neural network (SFNN) as an associative memory. Several simulations are provided to verify that not only our model needs much fewer neurons to store numerous stable fixed points, but also it is able to learn asymmetric arrangement of fixed points, whereas the SFNN model is limited to orthogonal arrangements.

Keywords: Feedback auto-associative memory; Stability analysis; Fixed points

1. Introduction

Utilized in a wide spectrum of applications, neural networks have been widely studied in recent years [8,15,17]. In the conventional structure of an artificial neural network, a neuron receives its input either from other neurons or from external inputs (input vector). A weighted sum of these inputs constitutes the argument of a fixed nonlinear activation function. The resulting value of the activation function is the neural output [10].

The associative neural networks (AsNNs) are dynamical nonlinear systems capable of processing information via the evolution of their states in a high dimensional state-space [7]. They are evolved from networks of interconnected neurons. This subject has received most research attention after the study of Amari [1] and Hopfield [11], which revealed how this kind of neural networks possesses associative properties. There are 2 main requirements for an AsNN: First, every given memory must be an equilibrium point of the network. Furthermore, the equilibria corresponding to the memory vectors have to be asymptotically stable, i.e., they should be attractive equilibrium points or attractors [3].

AsNNs store a set of desired patterns as stable equilibrium points such that the stored information can be retrieved if sufficient data is provided in an input pattern. In other words, AsNNs provide distributed storage of information, within which every neuron stores fragments of information needed to retrieve any stored data record. This property of AsNNs makes them suitable for various applications. Having high generalization and error-correction capabilities, they can be used as efficient noise-removing filters or distortion-tolerant tools [7]. The generalization abilities of AsNNs can be exploited in many classification tasks, such as image segmentation [6] and chemical substances recognition [16].

In general, desired memory patterns are presented by binary or real-valued (analog) vectors [4,5]. Hopfield [11] presented continuous-time feedback neural networks, which provided a way of storing analog patterns. Storage of analog pattern vectors with real-valued components is of
great interest; since in applications such as pattern recognition, vector quantization and image processing, the patterns are originally in analog form and costly quantization operation may be avoided [12,19]. Another method for storing analog patterns in Hopfield recurrent associative memories is introduced in [3], which points out some of the limitations of the Hopfield continuous-time model. In [18], some mathematical properties of labeling recursive auto-associative memories are discussed, and some conditions on the asymptotical stability of the decoding process along a cycle of the encoded structure are given. An example for the application of feedback associative memories is presented in [20], where binary vectors represent the digit inscriptions to be stored in the network.

A simple auto-associative memory model for storing analog and digital patterns is recently presented in [2], which is based on a self-feedback neural network (SFNN). The SFNN is a 2-layer recurrent network (Fig. 1) in which the output layer contains self-feedback neurons. The self-feedback connection of neurons ensures that the output of the SFNN contains the complete past information of the system even if the inputs of the SFNN are the present states of the system. Since there is no interlink between self-feedback neurons in the output layer, the SFNN has considerably fewer weights than fully recurrent neural networks (FRNN) and the network is noticeably simplified [13].

In this paper, a feedback auto-associative memory comprised of a simple feed-forward neural network (multi-layer perceptron or MLP), is presented to store real-valued patterns. To this end, the feed-forward portion of the system is recruited as an auto-associative memory. In other words, the desired fixed points are instructed to the MLP both as inputs and desired outputs. The feedback connection is then established using delay elements. We investigate the dynamical behavior of the resulting structure, which is a feedback auto-associative memory.

The outline of the paper is as follows: In Section 2, we review some important concepts regarding dynamical systems, and then we introduce the structure of the network. In Section 3, based on dynamical systems theory, we derive some explicit equations which relate the stability properties of fixed points to the network parameter values. Simulation results are provided in Section 4 in order to demonstrate the accuracy of the presented mathematical deductions. The performance comparison between the proposed model and the SFNN as an auto-associative memory model is presented in Section 5. Finally, Section 6 concludes the paper.

2. Methods

2.1. Preliminaries

In general, dynamical systems may be discrete or continuous, depending on whether they are described by differential or difference equations. The difference equation for a general time-invariant discrete dynamical system can be written as

$$X_{k+1} = f(X_k), \quad k = 0, 1, \ldots,$$

where $X \in \mathbb{R}^n$, and $f : \mathbb{R}^n \to \mathbb{R}^n$ can be a linear or nonlinear function of $X_k$. Considering (1), we have the following definitions and theorems [14]:

**Definition 1.** A point $\bar{x} \in \mathbb{R}$ is an equilibrium point for the dynamical system (1), or a fixed point for the map $f$, if $f(\bar{x}) = \bar{x}$.

**Definition 2a.** A fixed point $\bar{x}$ of (1) is said to be stable if for any $\varepsilon > 0$ there exists $\delta > 0$ such that whenever $|x_0 - \bar{x}| < \delta$, the point $\bar{x}$ satisfies $|x_k - \bar{x}| < \varepsilon$.

**Definition 2b.** A fixed point $\bar{x}$ of (1) is said to be unstable if it is not stable.

**Definition 2c.** A fixed point $\bar{x}$ of (1) is said to be asymptotically stable or a sink or an attracting fixed point of the function $f$ if it is stable and, in addition, there exists $r > 0$ such that for all $x_0$ satisfying $|x_0 - \bar{x}| < r$, the sequence $x_k$ satisfies $\lim_{k \to \infty} x_k = \bar{x}$.

**Theorem 1.** Let $f : \mathbb{R} \to \mathbb{R}$ be continuously differentiable in a neighborhood of the fixed point $\bar{x}$. Then $\bar{x}$ is asymptotically stable if $|f'(\bar{x})| < 1$ and is unstable if $|f'(\bar{x})| > 1$.

**Theorem 2.** Let $f : \mathbb{R}^n \to \mathbb{R}^n$ and $\bar{X}$ be a fixed point of (1). If the eigenvalues of the Jacobian matrix, all have absolute values less than “1”, then $\bar{X}$ is an asymptotically stable fixed point. If one of the eigenvalues of the Jacobian matrix has an absolute value greater than “1”, then $\bar{X}$ is an unstable fixed point.

2.2. The structure of the network

The proposed structure, as illustrated in Fig. 2, is composed of a 3-layer perceptron and a feedback connection.
which connects the network output to its input through delay elements. Obviously, the input and the output vectors have the same number of components. A sigmoid transfer function is chosen for the nodes of the hidden layer; while at the output layer there is a linear summation. The network output is formulated as:

$$X_{k+1} = f(X_k) = Vg(WX_k + B_1) + B_2,$$  \hspace{1cm} (2)

where $g(\lambda) = 1/(1 + \exp(-\lambda))$ is the activation function, $W$ and $B_1$ are weight and bias matrices linking the input layer to the hidden layer, and $V$ and $B_2$ are weight and bias matrices linking the hidden layer to the output layer.

At the training stage, the MLP is exploited as an auto-associative memory; i.e., the desired fixed points are randomly instructed to the MLP so that the input and the output of the network are equal at these points. This is accomplished by starting from random initial network weights, giving fixed points as both inputs and desired outputs to the MLP, back-propagating the output mean square error (MSE) into the network connections, and finally, modifying the network weights in order to reduce the MSE. The training stage continues until a sufficiently low MSE is achieved.

The feedback connection is established after the training stage. Next, we start from an initial point and calculate the network output recursively using (2). The algorithm continues until the stop criterion is satisfied, i.e., when the Euclidean distance between 2 sequential outputs is smaller than a predefined threshold.

3. Stability analysis

In this section, we analyze the stability properties of fixed points in the presented structure.

3.1. One-dimensional (1D) systems

In a 1D structure, we can rewrite (2) in an expanded form, as:

$$x_{k+1} = f(x_k) = \sum_{i=1}^{n} v_ig(w_ix_k + b_{i1}) + b_2,$$  \hspace{1cm} (3)

where $n$ is the number of the hidden layer neurons, and the other parameters are as defined in (2). Using Definition 1, at the fixed point $\bar{x}$ we have:

$$x_{k+1} = x_k = \bar{x} \Rightarrow \sum_{i=1}^{n} v_ig(w_ix + b_{i1}) + b_2 = \bar{x}. \hspace{1cm} (4)$$

A geometrical solution to (4), for $n = 3$, is illustrated in Fig. 3, assuming $y_1 = x$ and $y_2 = \sum_{i=1}^{3} v_ig(w_i\bar{x} + b_{i1}) + b_2$. $g(x)$ is the activation function as defined in Section 2.2. The intersections between the 2 curves ($y_1$ and $y_2$) demonstrate the position of the fixed points of the dynamical system. Using Fig. 3, we can also investigate the stability properties of the fixed points: according to Theorem 1, when the slope of $y_2$ is smaller than the slope of $y_1$ (equal to “1”) the fixed point is asymptotically stable (marked by filled circles). On the other side, when the slope of $y_2$ is greater than the slope of $y_1$, the fixed point is unstable (marked by empty circles).

The derivative of $f(x)$ can be obtained using (3):

$$f'(\bar{x}) = \sum_{i=1}^{n} v_ig'(w_i\bar{x} + b_{i1}) = \sum_{i=1}^{n} v_ig'(w_i\bar{x} + b_{i1})[1-g(w_i\bar{x} + b_{i1})]. \hspace{1cm} (5)$$

Using (5) and Theorem 1, we are able to recognize which fixed point is asymptotically stable and which one is unstable. In fact, this equation relates the stability of fixed points to the network parameter values. However, this will require the calculation of the sigmoid function $g(x)$. For the case $n = 1$, we can rewrite (5) as:

$$f'(\bar{x}) = vw_ig'(\bar{x} + b_1)[1-g(\bar{x} + b_1)]. \hspace{1cm} (6)$$

Fig. 3. A geometrical solution to (4) for $n = 3$. 

Please cite this article as: M. Amiri, et al., Analysis of the dynamical behavior of a feedback auto-associative memory, Neurocomputing (2007), doi:10.1016/j.neucom.2007.07.027
By substituting \( g() \) from (4) into (6), after some calculations, we have:

\[
\begin{align*}
  f'(\tilde{x}) &= w(\tilde{x} - b)\left[1 - \frac{\tilde{x} - b}{v}\right], \\
  f'(x) &= -\frac{w}{v}x^2 + \frac{w}{v}(2b + v)x - \frac{w}{v}(b^2 + b)v.
\end{align*}
\]  

Eq. (7) points out a simpler way for calculating \( f(x) \).

### 3.2. Two-dimensional (2D) systems

In a 2D structure, we can rewrite (2) as:

\[
X_{k+1} = f(X_k) = Vg(WX_k + B_1) + B_2,
\]

where

\[
W = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix}, \quad V = \begin{bmatrix} v_{11} & v_{12} \\ v_{21} & v_{22} \end{bmatrix}, \quad B_1 = \begin{bmatrix} b_{11} \\ b_{21} \end{bmatrix}
\]

and

\[
B_2 = \begin{bmatrix} b_{12} \\ b_{22} \end{bmatrix}.
\]

The expanded form of (8) can be written as

\[
\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} v_{11}g(w_{11}x_1 + w_{12}x_2 + b_{11}) + v_{12}g(w_{21}x_1 + w_{22}x_2 + b_{21}) + b_{12} \\ v_{21}g(w_{11}x_1 + w_{12}x_2 + b_{11}) + v_{22}g(w_{21}x_1 + w_{22}x_2 + b_{21}) + b_{22} \end{bmatrix}.
\]

To obtain the fixed points of (9) based on Definition 1, we write:

\[
\begin{align*}
  \tilde{x}_1 &= f_1(\tilde{x}_1, \tilde{x}_2) \\
  &= v_{11}g(w_{11}\tilde{x}_1 + w_{12}\tilde{x}_2 + b_{11}) + v_{12}g(w_{21}\tilde{x}_1 + w_{22}\tilde{x}_2 + b_{21}) + b_{12}, \\
  \tilde{x}_2 &= f_2(\tilde{x}_1, \tilde{x}_2) \\
  &= v_{21}g(w_{11}\tilde{x}_1 + w_{12}\tilde{x}_2 + b_{11}) + v_{22}g(w_{21}\tilde{x}_1 + w_{22}\tilde{x}_2 + b_{21}) + b_{22}.
\end{align*}
\]

### Table 1

<table>
<thead>
<tr>
<th>Equilibrium points</th>
<th>Derivatives</th>
<th>Network parameter values</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( x_1 )</td>
<td>( \frac{dx_1}{dx_1} )</td>
</tr>
<tr>
<td>( x_1 = 0.1 )</td>
<td>0.553</td>
<td>-16.088</td>
</tr>
<tr>
<td>( x_2 = 0.2 )</td>
<td>1.456</td>
<td></td>
</tr>
<tr>
<td>( x_3 = 0.45 )</td>
<td>0.202</td>
<td></td>
</tr>
<tr>
<td>( x_4 = 0.2 )</td>
<td>0.380</td>
<td></td>
</tr>
<tr>
<td>( x_5 = 0.3 )</td>
<td>1.478</td>
<td></td>
</tr>
<tr>
<td>( x_6 = 0.4 )</td>
<td>0.608</td>
<td></td>
</tr>
<tr>
<td>( x_7 = 0.5 )</td>
<td>1.447</td>
<td>[29.20]</td>
</tr>
<tr>
<td>( x_8 = 0.6 )</td>
<td>0.580</td>
<td>[-27.91]</td>
</tr>
<tr>
<td>( x_9 = 0.7 )</td>
<td>1.546</td>
<td>27.13</td>
</tr>
<tr>
<td>( x_{10} = 0.8 )</td>
<td>0.333</td>
<td></td>
</tr>
</tbody>
</table>

For the case that the weight matrices are anti-diagonal, we can calculate the eigenvalues as:

\[
\tilde{\lambda}_1 = v_{11}w_{11}g(w_{11}\tilde{x}_1 + w_{12}\tilde{x}_2 + b_{11}) + v_{12}w_{21}g(w_{21}\tilde{x}_1 + w_{22}\tilde{x}_2 + b_{21}) + b_{11},
\]

\[
\tilde{\lambda}_2 = v_{12}w_{22}g(w_{22}\tilde{x}_2 + b_{21}),
\]

\[
\tilde{\lambda}_3 = v_{21}w_{21}g(w_{21}\tilde{x}_1 + w_{22}\tilde{x}_2 + b_{21}) + v_{22}w_{22}g(w_{22}\tilde{x}_2 + b_{21}) + b_{21}.
\]

Next, we calculate the 2 \times 2 Jacobian matrix, \( J \), in order to investigate the stability properties of the fixed points:

\[
J = \begin{bmatrix} v_{11}w_{11}g(w_{11}\tilde{x}_1 + w_{12}\tilde{x}_2 + b_{11}) + v_{12}w_{21}g(w_{21}\tilde{x}_1 + w_{22}\tilde{x}_2 + b_{21}) \\ v_{12}w_{22}g(w_{22}\tilde{x}_2 + b_{21}) \\ v_{21}w_{21}g(w_{21}\tilde{x}_1 + w_{22}\tilde{x}_2 + b_{21}) + v_{22}w_{22}g(w_{22}\tilde{x}_2 + b_{21}) \end{bmatrix}.
\]

Using the eigenvalues of (11) and based on Theorem 2, we are able to distinguish between asymptotically stable and unstable fixed points in the 2D structure.

For the case that the weight matrices are diagonal, we can simply calculate the eigenvalues as:

\[
\tilde{\lambda}_1 = v_{11}w_{11}g(w_{11}\tilde{x}_1 + b_{11}) + v_{12}w_{21}g(w_{21}\tilde{x}_1 + w_{22}\tilde{x}_2 + b_{21}),
\]

\[
\tilde{\lambda}_2 = v_{12}w_{22}g(w_{22}\tilde{x}_2 + b_{21}),
\]

\[
\tilde{\lambda}_3 = v_{21}w_{21}g(w_{21}\tilde{x}_1) + v_{22}w_{22}g(w_{22}\tilde{x}_2 + b_{21}) + b_{21}.
\]

\[
\tilde{\lambda}_4 = v_{11}w_{11}g(w_{11}\tilde{x}_1 + w_{12}\tilde{x}_2 + b_{11}),
\]

\[
\tilde{\lambda}_5 = v_{12}w_{21}g(w_{21}\tilde{x}_1 + w_{22}\tilde{x}_2 + b_{21}),
\]

\[
\tilde{\lambda}_6 = v_{21}w_{22}g(w_{22}\tilde{x}_2 + b_{21}),
\]

\[
\tilde{\lambda}_7 = v_{22}w_{21}g(w_{21}\tilde{x}_1 + w_{22}\tilde{x}_2 + b_{21}) + v_{22}w_{22}g(w_{22}\tilde{x}_2 + b_{21}) + b_{21}.
\]

For the case that the weight matrices are anti-diagonal, we can calculate the eigenvalues as:

\[
\tilde{\lambda}_1 = v_{11}w_{11}g(w_{11}\tilde{x}_1 + w_{12}\tilde{x}_2 + b_{11}),
\]

\[
\tilde{\lambda}_2 = v_{12}w_{21}g(w_{21}\tilde{x}_1 + w_{22}\tilde{x}_2 + b_{21}),
\]

\[
\tilde{\lambda}_3 = v_{21}w_{22}g(w_{22}\tilde{x}_2 + b_{21}),
\]

\[
\tilde{\lambda}_4 = v_{22}w_{21}g(w_{21}\tilde{x}_1 + w_{22}\tilde{x}_2 + b_{21}).
\]
and after some calculations:
\[
\begin{align*}
l_1 &= w_{21}(x_1 - b_{12}) - \frac{w_{21}}{v_{21}} (x_1 - b_{12})^2, \\
l_2 &= w_{12}(x_2 - b_{22}) - \frac{w_{12}}{v_{12}} (x_2 - b_{22})^2.
\end{align*}
\]

(15)

4. Experiments and results

In the following experiments, we train the feedback auto-associative memory in the way mentioned in Section 2.2. The Levenberg–Marquardt (LM) algorithm is used to train the network [9], and the final stopping MSE is set to $10^{-12}$. After training, we investigate the stability properties of fixed points by starting from an initial point and calculating the network output recursively using (2). The algorithm continues until the Euclidean distance between 2 sequential network outputs is less than $10^{-10}$.

4.1. One-dimensional structure

In the 1D structure, the network has 1 input node and 1 linear output neuron. In the first experiment, we train the network with 3 fixed points shown in Table 1 and Fig. 4(a). When the number of hidden layer neurons is 1, the network is able to learn the desired mapping function. After training, we establish the feedback connection in order to obtain the trajectories of the system. The trajectories, as sketched in Fig. 4(b), diverge from unstable and converge at stable fixed points. The filled circles show the learned fixed points.

![Graphs showing fixed points and phase portraits](image)

Fig. 5. (a) and (c) training fixed points; (b) and (d) resulting phase portrait. Each line indicates the path through which the network output approaches a stable fixed point, starting from an initial point in the two-dimensional state space.

Please cite this article as: M. Amiri, et al., Analysis of the dynamical behavior of a feedback auto-associative memory, Neurocomputing (2007), doi:10.1016/j.neucom.2007.07.027
We use (7) to calculate the derivative values at these fixed points, as given in Table 1. According to Theorem 1, the 2 lateral points \((x_1\) and \(x_3\)) are asymptotically stable \(\left| f'(x_1) \right| < 1\) and \(\left| f'(x_3) \right| < 1\) while the other point \((x_2)\) is unstable \(\left| f'(x_2) \right| < 1\).

In the second experiment we increase the number of fixed points to 7 (Fig. 4(c)). The experiment shows that the number of neurons in the hidden layer must be at least 3 in this case, in order to enable the network to store all 7 fixed points. The resulting phase portrait is depicted in Fig. 4(d).

4.2. Two-dimensional structure

In this experiment, we train the network with a \(3 \times 3\) matrix of fixed points (Fig. 5(a)). The MLP has 2 neurons in its hidden layer. After training, we investigate the stability properties of fixed points by exploiting the aforementioned approach. In Fig. 5(b), each line shows a path through which the network output approaches an asymptotically stable fixed point, starting from an initial point in the 2D state space. In the next experiment, we train the network with a \(5 \times 5\) matrix of fixed points (Fig. 5(c)). Our simulations demonstrate that the network is able to learn this data set when there are at least 4 neurons in its hidden layer. The final stopping MSE is \(10^{-12}\) and the resulting phase portrait is shown in Fig. 5(d).

Since unstable fixed points are specified by the training data, and because of their role in forming the boundaries of attraction domains, we are able to displace these boundaries. In the following experiment, some deviations are

Fig. 6. (a) and (b) training fixed points; (c) and (d) resulting phase portrait. Each line indicates the path through which the network output approaches a stable fixed point, starting from an initial point in the two-dimensional state space.

Please cite this article as: M. Amiri, et al., Analysis of the dynamical behavior of a feedback auto-associative memory, Neurocomputing (2007), doi:10.1016/j.neucom.2007.07.027
made to the $3 \times 3$ training data set, as shown in Figs. 6(a) and (b). In Fig. 6(a) global asymmetric shift is applied to the training data, while in Fig. 6(b), we have modified the data set by rotating the central axis. The resulting phase portrait is depicted in Figs. 6(c) and (d), respectively. As it is evident from these figures, the attraction domain of each asymptotically stable fixed point can be controlled by manipulating the training data.

For the data set shown in Figs. 5(a) and 6(a), we use (15) and (13), respectively, to calculate the eigenvalues of the Jacobian matrix. Table 2 demonstrates the calculated eigenvalues. This table clearly confirms the resulting phase portrait illustrated in Figs. 5(b) and 6(c); since according to Theorem 2, the fixed points with eigenvalues more than “1” (B, D, E, F, H) have to be unstable whereas the fixed points with eigenvalues less than “1” (A, C, G, I) must be asymptotically stable.

5. Performance comparison with SFNN

In this section, we perform some case studies for the purpose of performance comparison between the proposed model and other AsNN models. To this end, we replicate some of the previous simulations for an SFNN.
as an auto-associative memory. We use the data set shown in Figs. 5(a) and 6(a) to train the SFNN model. The training method for the SFNN is comprehensively investigated in [2]. Incorporating 2 neurons in its output layer, the SFNN is able to store the desired fixed points with a final MSE of $10^{-11}$. The resulting phase portraits are shown in Figs. 7(a) and (b), respectively. Comparing Figs. 7(a) and (b) with Figs. 5(b) and 6(c), there is not any considerable difference evident in the 2 results, and the 2 networks act in a similar way.

According to the SFNN model, each neuron can store at most 3 fixed points, 2 of which are asymptotically stable and the other is unstable [2]. Therefore, the SFNN cannot store more than 4 asymptotically stable fixed points when the dimension of the state space is 2. In order to store 8 stable fixed points in an SFNN structure, at least 3 output neurons are needed, that leads to the expansion of the state space dimension to 3. Nevertheless, the proposed model was able to store 9 stable fixed points in the 2D state space (see Fig. 5(d)).

Since the difference equations describing the SFNN model are uncoupled [2], the rotation of the central axis as shown in Fig. 6(b) is impossible. Instead, the fixed points must be arranged in an orthogonal configuration. This limitation restricts us in manipulating the attraction domain boundaries. Alternatively, our model was able to deal with such situations (see Fig. 6(d)).

6. Conclusion

Currently, AsNNs are among the most extensively studied and understood neural paradigms. Furthermore, storage of analog pattern vectors with real-valued components is of great interest. Associative memories can be exploited to store a set of desired patterns as stable equilibrium points such that the stored information can be retrieved. In this study, we investigated the dynamical behavior and the stability properties of fixed points in a feedback auto-associative memory. The training procedure was reduced to the point of training a feedforward neural network, and therefore the well-known backpropagation algorithm was used. We derived some explicit equations based on the theory of dynamical systems, which related the stability properties of fixed points to the network parameter values. In comparison to other associative memory models such as SFNN, our model has at least 2 significant advantages: First, our model needs much fewer neurons to store numerous stable fixed points. Second and more important, it is able to learn asymmetric arrangements of fixed points (see Fig. 6(d)) whereas SFNN is limited to orthogonal arrangements. In many applications such as nonlinear control of manipulators, having an adjustable attraction domain is of great importance.

Acknowledgement

The authors would like to thank the four anonymous reviewers for their valuable comments on earlier versions of this paper.

References

M. Amiri received his B.Sc. and M.Sc. degrees (Hons.) all in Biomedical Engineering from University of Isfahan, Isfahan, Iran in 2004, Amirkabir University of Technology (Tehran Polytechnic), Tehran, Iran, in 2006, respectively. He is currently studying his Ph.D. program in Biomedical Engineering. His main interests are in computational neuroscience, intelligent control, and theoretical and practical aspects of neural networks.

S. Saeb received his B.Sc. and M.Sc. degrees in Electrical Engineering and Biomedical Engineering, both from Amirkabir University of Technology (Tehran Polytechnic), Tehran, Iran, in 2002 and 2005, respectively. His research interests focus on neural networks, models of cortical synaptic plasticity, auditory signal processing, and dynamic synapse neural networks.

M. J. Yazdanpanah received his B.Sc., M.Sc., and Ph.D. degrees all in Electrical Engineering from Isfahan University of Technology, Isfahan, Iran in 1986, Tehran University, Tehran, Iran, in 1988, and Concordia University, Montreal, Quebec, Canada in 1997, respectively. His Ph.D. thesis entitled “Control of flexible-link manipulators using nonlinear $H_{\infty}$ techniques” was ranked outstanding.

From 1986 to 1992, he worked with the Engineering Research Center, Tehran, Iran, as an R&D engineer and culminating as the chairman of the System Design Division.

In 1998, he joined the School of Electrical and Computer Engineering, University of Tehran, Tehran, Iran, where he is now an Associate Professor, serving as instructor of various courses and also supervisor of M.Sc. and Ph.D. students.

Dr. Yazdanpanah’s research interests are in the areas of analysis and design of nonlinear/robust/optimal/adaptive control systems, robotics, control on networks, and theoretical and practical aspects of neural networks.

S. A. Seyyedsalehi received his B.Sc. and M.Sc. degrees both in Electrical Engineering, from Sharif University of Technology, Tehran, Iran in 1982, Amirkabir University of Technology (Tehran Polytechnic), Tehran, Iran, in 1989 and Ph.D. degree in Biomedical Engineering from Tarbiat Modarres University, Tehran, Iran in 1996.

Dr. Seyyedsalehi’s research interests are in the areas of speech processing and recognition, biological and artificial neural networks, neural modeling and linear and nonlinear signal processing. He is now an Assistant Professor in the Faculty of Biomedical Engineering, Amirkabir University of Technology.