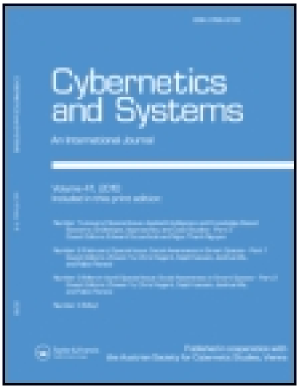


Cybernetics and Systems: An International Journal
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/ucbs20

ADAPTATION BASED ON MEMORY DYNAMICS IN A CHAOTIC NEURAL NETWORK

Nigel Crook & Tjeerd Olde Scheper

Version of record first published: 30 Nov 2010.

To cite this article: Nigel Crook & Tjeerd Olde Scheper (2002): ADAPTATION BASED ON MEMORY DYNAMICS IN A CHAOTIC NEURAL NETWORK, Cybernetics and Systems: An International Journal, 33:4, 341-378

To link to this article: http://dx.doi.org/10.1080/01969720290040632

Cybernetics and Systems: An International Journal, 33: 341–378, 2002
Copyright © 2002 Taylor & Francis
0196-9722/02 $12.00 + .00
DOI: 10.1080/01969720290040632

ADAPTATION BASED ON MEMORY DYNAMICS IN A CHAOTIC NEURAL NETWORK

NIGEL CROOK
TJEERD OLDE SCHEPER

Neurocomputing Research Group, School of Computing and Mathematical Sciences, Oxford Brookes University, Oxford, UK

The complex dynamics that emerge from systems governed by deterministic chaos offer significant advantages to the neuromorphic engineer. Included in these is the potential for a very large memory store and the ease with which chaotic systems can be controlled. By definition, a chaotic system is aperiodic. However, during the course of its trajectory through state space, the chaotic system will come infinitely close to points that it has previously visited. These almost repeating trajectories are referred to as Unstable Periodic Orbits (UPOs). Normally, under the influence of chaos, the trajectory would move away exponentially fast from its previous path, thereby describing a new path on the surface of the attractor. It is possible to apply a simple delayed feedback control mechanism to a chaotic system that will constrain it within one of its UPOs. This article presents a neural implementation of this delayed feedback mechanism. The network presented here is able to stabilize different UPOs in response to different input signals, with each UPO corresponding to a dynamic recognition state for that input. We also present two learning rules for this network, which enables it to adapt to novel inputs in a self-organized manner.

INTRODUCTION

The research presented in this paper attempts to identify and model ways to store information in dynamic, chaotic neural networks. The justification for this research is given by both biological as well as theoretical

Address correspondence to Nigel Crook, Intelligent Systems Research Group, School of Computing and Mathematical Sciences, Oxford Brookes University, Wheatley Campus, Oxford OX33 1HX, UK. E-mail: ntcrook@brookes.ac.uk

motivations (Aihara, Takabe, and Toyoda 1990; Freeman 1985; Freeman 1987; Freeman and Barrie 1994; Kaneko and Tsuda 1994; Tsuda 1996). Firstly, there seems to be substantial support for the use of dynamic networks to study more complex and interesting behavior. Artificial neural networks (ANN) have specific properties that define their order, such as size, type, and function. Simply extending the ANN with complex non-linear dynamics does not improve the memory performance of the network. It may, however, modify the rate at which a global minimum is located, if such a state exists. Using non-linear differential equations may add more complexity to the system and thereby increase the possible memory states. Secondly, even though chaos may seem to be generally undesirable, it has important properties that may be exploited to store and retrieve information (Sinha and Ditto 1999). These are the space filling, the possibility of control via delayed feedback, synchronization and the sensitive dependence on initial conditions. In this paper we demonstrate how these unique properties may be exploited to store information in the dynamic behavior of a neural network. Furthermore, we present a novel approach to neural network adaptation which is based on supporting the dynamics from which memory states emerge during pattern recognition.

CHAOS

The definition of chaos is complex but is well described in the case of models. Signal analysis of a chaotic signal is even more complex and the identification of chaos is only assured in specific types of signals. The difference between noisy and chaotic signals may be lost if a signal is indistinguishable from a random noise. There exist, however, some measures which may indicate the possibility of chaos in the case of low-dimensional chaotic signals. High-dimensional chaos is often too complex to enable a numerical algorithm to separate it from noise. The attracting set of a dissipating chaotic system or chaotic attractor is usually referred to as a strange attractors because of the fractal dimension of the attractor, i.e. the attractor has a dimension that is not an integer. The right panel (*b*) of Figure 1 shows the strange attractor of the Rössler equation (Rössler 1976):

$$\dot{x} = -y - z \tag{1}$$

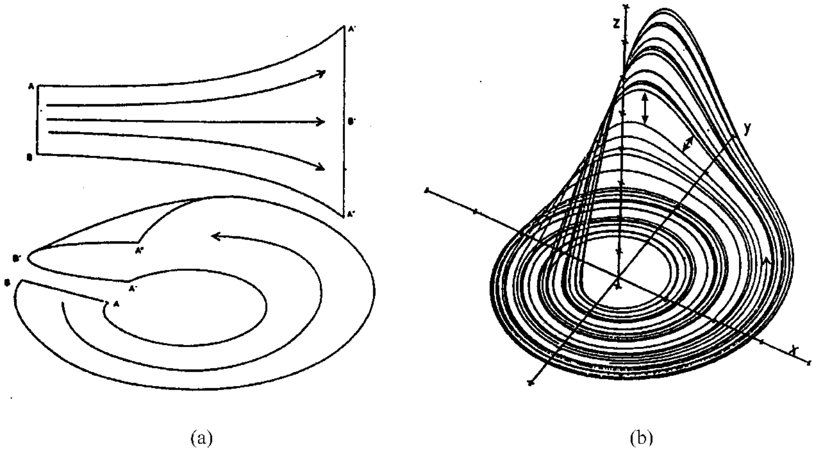


Figure 1. Stretching and folding (or contraction) of an unstable manifold, three close by trajectories diverge with exponential rate. Folding of a manifold with the direction of evolution: *(b)* The Rössler attractor with both stretching and folding.

$$\dot{y} = x + \alpha y \tag{2}$$

$$\dot{z} = \beta + (xz) + \gamma z \tag{3}$$

with $\alpha = 0.2$, $\beta = 0.2$, and $\gamma = -5.7$. In the right panel (*b*) of Figure 1 at the top of the z-axis the stretching of the attractor can be seen, this is demonstrated by two trajectories first close together but later expanding away as shown. At the x, y plane the folding of the trajectory back into the z-direction is recognizable. This is also indicated in the left panel (*a*) in the same figure. Here the upper plot demonstrates the exponential rate of expansion of three trajectories. The lower plot demonstrates the folding of trajectories. Another property of a chaotic system is its dependence on initial conditions. With a very small difference in initial conditions, two identical chaotic systems may diverge away from each other at an exponential rate. The shape of the attractor will remain the same, but the two systems will traverse the phase space differently. This is shown in Figure 2 where two trajectories of the variable x are depicted with the same parameter values but with different initial conditions. For one trajectory the initial condition is $x = 5$ and for the other $x = 5.0001$. The initial values of y and z are the same in both cases. (Note that even

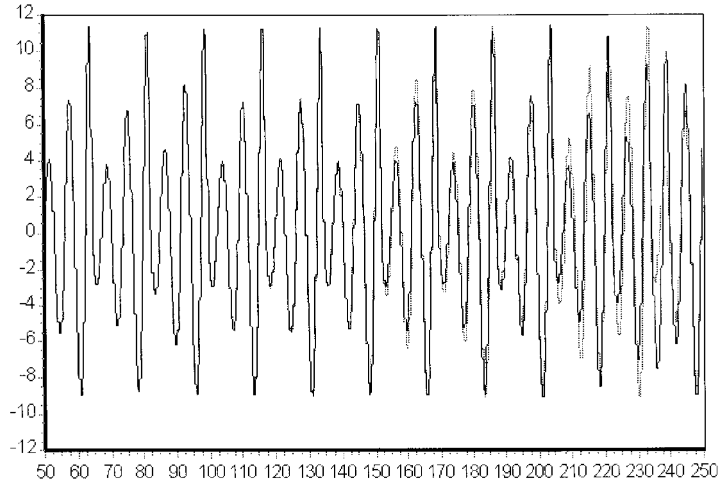


Figure 2. Dependence on initial conditions of the Rössler model, numerical divergence exists almost instantaneously, but becomes visible at approximately time step 135.

though the difference is still fairly large, i.e. 0.01%, the sensitive dependence on initial conditions is valid, but with a smaller initial difference between the two trajectories, the divergence would take much longer before the effect becomes visible.)

Chaos is often considered to be an undesirable property of systems because it tends to complicate analysis and function. However, making large modifications to a system parameter to reduce chaotic behavior may be undesirable since it may well change the behavior of the system in unacceptable ways. Rather than having to redefine the system, in most cases the presence of chaos makes it is possible to change the behavior of the system with only small changes to one of the system parameters. Typically, most chaotic attractors embed a large, dense set of unstable periodic orbits (UPO). This is shown in Figure 3 where an unstable periodic orbit is indicated inside the strange attractor described by the Rössler system.

By determining several of these orbits and selecting the orbit that improves system performance a chaotic system may be controlled. A specific unstable periodic orbit may be stabilized by making appropriate pre-programmed modifications to the parameter. If the changes are small enough the orbits will not have completely different properties than

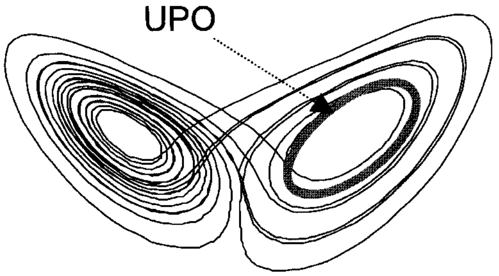


Figure 3. An unstable periodic orbit in the Rössler attractor.

the unchanged attractor. The stabilization of an UPO in this way is called the control of chaos.

One method of controlling chaos requires making a *small* time-dependent perturbation to a controllable system parameter. This has first been numerically demonstrated by Ott (1993); Schuster (1999); and Ott, Grebogi, and Yorke (1990) and is referred to as the OGY method of chaos control. A different method that has been used successfully is delayed feedback control of a chaotic system (Pyragas 1995; olde Scheper 2001). The continuous feedback method for controlling chaos presented by Pyragas (1992) has been applied to continuous time systems. This method of controlling chaos assumes that a continuous time system has an output variable, say $y(t)$, that can be measured and an input signal, $F(t)$:

$$\begin{aligned} \frac{dy}{dt} &= P(y, \mathbf{x}) + F(t) \\ \frac{d\mathbf{x}}{dt} &= \mathbf{Q}(y, \mathbf{x}) \end{aligned} \tag{4}$$

Here, $P(y, x)$ and $\mathbf{Q}(y, \mathbf{x})$, which govern the chaotic dynamics of the system, and the matrix $\mathbf{x}$, which denotes all of the remaining system variables, are assumed to be unknown. When the control signal $F(t)$ is zero, the system (4) is governed by a chaotic attractor. The input signal $F(t)$ is proportional to the difference between the value of y at time t and the value of y at time $t - \tau$, where τ is a fixed delay:

$$F(t) = K[y(t) - y(t - \tau)] \tag{5}$$

The input signal $F(t)$ attempts to nudge the system back to a state in which output variable y repeats the same value it had at the earlier time specified by the delay τ. In this way, $F(t)$ encourages the system to follow a periodic trajectory with periodicity τ (see Figure 4).

As the system approaches the periodic trajectory, $F(t)$ will become very small. Figure 5 shows a time series for the Rössler system. The controlling input signal is initially zero and the system follows its chaotic attractor for a period of time. When the input signal is activated the system quickly converges to a period one UPO. Figure 5 shows a burst of activity in $F(t)$ which the system is brought under control. Subsequently, as the system moves into the UPO, $F(t)$ becomes very small.

The same method can also be applied to discrete time systems. All that is required is that the system has a measurable output variable $y(t)$ and a controlled input signal $F(t)$:

$$\begin{aligned} G(t+1) &= P(y(t), \mathbf{x}(t)) \\ y(t) &= G(t) + F(t) \\ \mathbf{x}(t+1) &= \mathbf{Q}(y(t), \mathbf{x}(t)) \\ F(t) &= k[G(t) - y(t-\tau)] \end{aligned} \tag{6}$$

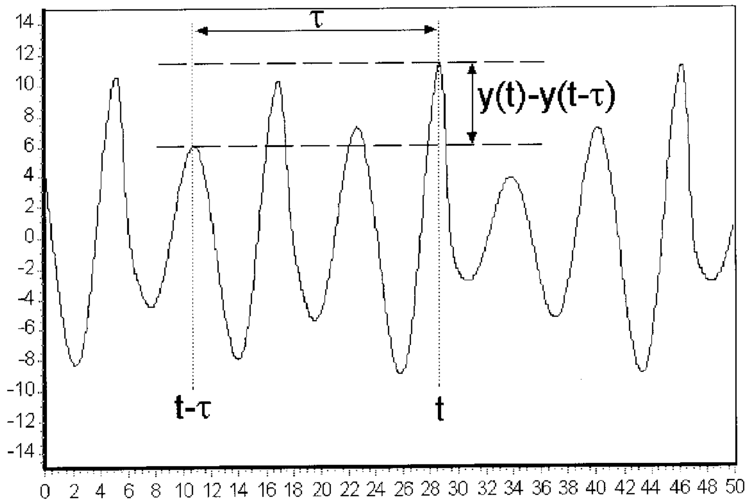


Figure 4. A sample time series in y with the time delay τ superimposed.

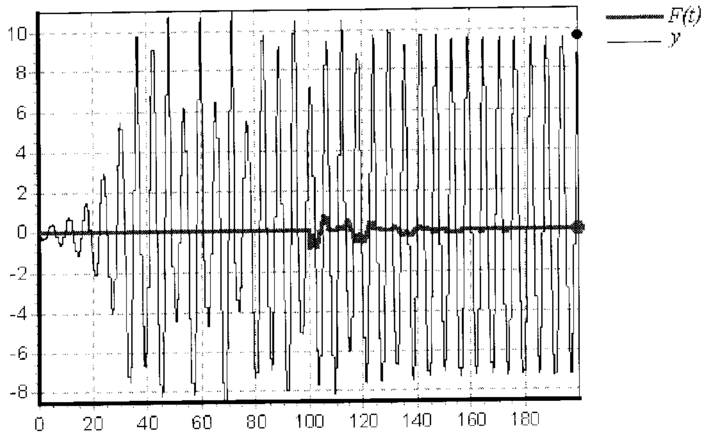


Figure 5. Time series plot for one of the Rössler equation variables, superimposed on a plot of $F(t)$.

where $P(y(t), x(t))$ and $\mathbf{Q}(y(t), \mathbf{x}(t))$ govern the chaotic dynamics of the system, and $\mathbf{x}$ denotes all of the remaining system variables. The mechanism of stabilization is similar to the continuous method, but the control compares the new point in $G(t)$ to the delayed point $y(t-\tau)$, instead of the current point.

By using delayed feedback with varying parameters, it is possible to select a specific orbit when presenting the system with periodic input. This is proposed to be the "recognized state," while the normal chaotic state is called the "undetermined state." The existence of an UPO may be influenced by adapting the parameter space. Thus by making small changes to a selected parameter, UPOs may be found, depending on the period of the external input (Crook, Dobbyn, and olde Scheper 2000; Tsui and Jones 1999). In a locally connected network of delayed controlled neurons it is possible to demonstrate the control response of responsive units in specific regions. Units that are capable of controlling a specific UPO will stabilize into the UPO when presented with associated input.

CHAOTIC NETWORK MODELS

This section describes some interesting models proposed by different authors, that have various approaches to modelling a chaotic neural network with control. A very useful model is proposed by Aihara, Takabe, and

Toyoda (1990), that describes a single neuron model with chaotic dynamics, including graded responses, relative refractoriness and spatio-temporal summation of inputs. The model is derived from the Caianiello's neuronic equation, of which the McCulloch-Pitts neuron is a special case. A modified version of the same equation was used by Nagumo and Sato to develop the following model to produce the output $x(t+1)$:

$$x(t+1) = u\left(A(t) - \alpha \sum_{r=0}^{t} k^r x(t-r) - \theta\right) \tag{7}$$

where u is a unit step function or sigmoid function, $A(t)$ is the input strength at time t and k is the damping of the refractor rate. Aihara et al. defined a new internal state $y(t+1)$ as:

$$y(t+1) = A(t) - \alpha \sum_{r=0}^{t} k^r x(t-r) - \theta \tag{8}$$

subsequently, (7) and (8) can then be simplified into:

$$y(t+1) = ky(t) - \alpha u(y(t)) + a(t) \tag{9}$$

$$x(t+1) = u(y(t+1)) \tag{10}$$

$$\text{where} \quad a(t) = A(t) - kA(t-1) - \theta(1-k) \tag{11}$$

If the input into the network is periodic with constant amplitude A then (11) may be used as $a = (A - \theta)(1 - k)$. The response characteristic of the equations (9) and (10) form complete devil's staircases, this means that chaotic solutions exist only at a self-similar Cantor set of the parameter values with zero Lebesgue measure. It is shown by Aihara et al. that this model may be used to produce a chaotic neural network with a small positive largest Lyapunov exponent.

The Aihara model has been used by Kushibe, Liu, and Othsubo (1996) to build a network that can target a specific embedded memory by associating it with a desired output. The incomplete target enables the system to locate a memory by chaotic searching avoiding local minima. Even when random input patterns are used, the system does not fall into a local minimum but will reproduce a memory pattern provided that at least 20% of the target pattern is available.

Mappings are often used in modeling discrete chaotic neural dynamics. A good example of this type of model is a map model by Pasemann and Stollenwerk (1998) that is described by a recurrent two-neuron model as:

$$x_{n+1} = \vartheta_1 + w_{11}\sigma(x_n) + w_{12}\sigma(y_n) \tag{12}$$

$$y_{n+1} = \vartheta_2 + w_{21}\sigma(x_n) + w_{22}\sigma(y_n) \tag{13}$$

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{14}$$

where x_n represents the state of an inhibitory neuron and y_n the state of an excitatory neuron. The variables are set as $\vartheta_1 = -2, \vartheta_2 = 3$, $w_{11} = -20, w_{12} = -6, w_{12} = 6$ and $w_2 2 = 0$, i.e. no auto inhibition of y_n. Applying a modified OGY form of control using linear least squares estimations of Δx_{n+k} on the parameter ϑ_1, this model may stabilize different periodic orbits with the current parameter set. Further modifications are made to establish a process of self-control, this comprises of four control neurons used as a one-layer feedforward network. The combination of the model and the control network allow stabilization of all available periodic orbits. The authors then use external or dynamic noise to switch between different orbits.

A comparable model as the Pasemann system is used by Klotz and Bräuer (1999). The system is described by:

$$x_i(t+1) = f\left(\sum_{j=1}^{n} w_{ij} x_j(t)\right) \tag{15}$$

$$f(z) = \frac{1}{1 + e^{-4\sigma z}} \tag{16}$$

From this several different network configurations can be constructed. A two-neuron model with appropriate weights demonstrates two chaotic attractors. A four-neuron model with time delay is constructed to show one chaotic attractor. Introducing an external input on one of the neurons reduces the system to one half of the attractor depending on the sign of the input. This model is then used by the authors to create an XOR function network. It shows chaotic behavior when the two inputs are both zero or one. It will reduce to one of the two halves of the attractor when one of the two inputs is one.

As in the previous model, instead of stabilizing a periodic orbit or state a particular subspace within the attractor or even different attractors may be used as an encoding of information. Itinerating a chaotic state among different attractors is used in a large model by Hoshino, et al. (1997). This model is a network of a three-neuron module as described by Chapeau-Blondeau and Chauvet (1992). This system is capable or chaotic and periodic oscillations within a certain distance of unit firing. This may then be modified by parameter changes in the system to represent different learning patterns. The system is not controlled but merely illustrates the ability to associate information with specific chaotic or periodic behavior in the parameter domain.

A compartmental neuronal model approach has been used by, Destexhe (1994) to model a network of delayed neurons. Within certain boundaries of the critical parameter values, particularly the weights, periodic oscillations, spiral waves, and spatiotemporal chaos was found. None of these dynamics can be produced without the delay, although the system is not particularly sensitive to the delay length. It is argued that the spatiotemporal phenomenon observed is useful to optimize information transfer within the network, even though the model does not demonstrate this.

A recurrent neural network model has been constructed by Andreyev, et al. (1996) that is trained to store a piecewise linear map. The exact nature of the map, i.e. the transition from one map state to another, is defined by the encoding of the desired information. This will result for n pieces of information of a repeated information block of length $n+1$. After training of the network with different blocks of information, the trajectory of the system will visit all regions of the stored patterns during iteration. Presentation of an input pattern will then stabilize a particular period associated with one of the information blocks. This model shows a useful method of encoding information temporally into a neural net that then may be searched chaotically for the correct pattern.

In papers by (Watanabe and Aihara (1997); Watanabe, Aihara, and Kondo (1998) the effect of coincidence detection in a network is investigated. A simple continuous neuron model is described with an exponential decay and threshold. The internal state is described as: $a(t_{n+1}) = s_{n+1} + a(t_n)e^{\frac{t_{n+1}}{\tau}}$ and a global negative feedback is calculated to reduce or increase the threshold value. This model can demonstrate coincidental firing of neurons in an almost synchronized manner.

A more complex neural network model with coincidence detector is also described. This second model is continuous and includes a time delay and exponential decay of the activation. A global negative feedback with time decay τ_g is applied to the network. The model is described as:

$$\tau \frac{da_i(t)}{dt} = -a_i(t) + S^{\text{ext}} \sum_n \delta(t - t^p_{i,n}) + \sum_{j=1}^{N} w_{ij} x_j(t - d_{ij}) \tag{17}$$

$$\tau_g \frac{dr(t)}{dt} = -r(t) + R \sum_{j=1}^{N} x_j(t) \tag{18}$$

If:

$$h(t) \equiv a_i(t) - (\theta + r(t)) \geq 0 \tag{19}$$

then:

$$x_i(t') = \delta(t' - g(h(t)) - t) \quad \text{and} \quad a_i(t') = 0 \tag{20}$$

otherwise:

$$x_i(t) = 0 \tag{21}$$

where S^{ext} is an external input pulse strength, R is the global feedback strength, $x_i(t)$ is the output of neuron i at t, $t^P_{i,n}$ is the time for external pulse n to arrive at neuron i, w_{ij} and d_{ij} are the synaptic weight and delay from neuron j to neuron i, and N the total number of neurons. All synaptic weights are set to 1 with the exception of autoconnections $w_{ii} = 0$. The external pulse is required for the activity of the network to be maintained. If the external pulse generation is increased over time the system will become unstable, due to the fact that the monotone increase of the pulse does not allow the system to stabilize onto a coinciding pulse. If the pulse is varied but not monotone, coincidental pulses may be found and the system becomes periodic.

NETWORK ARCHITECTURE

The principle that UPOs corresponding to memory states can be stabilized in response to specific input patterns is investigated in this section. Specifically, we present a neural implementation of the continuous delayed feedback method of chaos control (Pyragas 1992). This implementation is defined by a set of discrete-time equations which a three-model layered network. The first layer is composed of units which receive dynamic input signals from the *environment*. This input layer is fully connected to the second layer which is made up of inhibitory units. Each unit in the inhibitory layer is connected to one chaotic unit in the third layer. Units in the chaotic layer are connected to their immediate neighbors via lateral connections. An overview of the network architecture is shown in Figure 6.

The chaotic dynamics of each unit in the third layer are governed by the following discrete time equation which has been modified from Aihara, Takabe, and Toyoda (1990):

$$x(t+1) = \omega x(t) - \alpha f(x(t)) + a \tag{22}$$

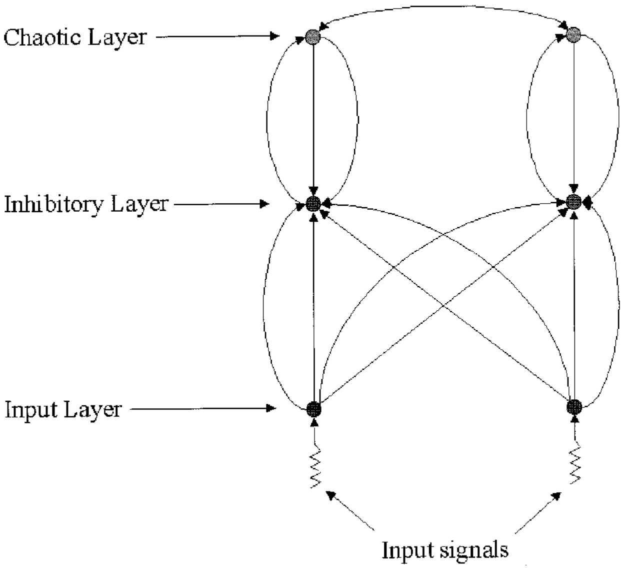


Figure 6. An overview of the network architecture.

where $x_i(t)$ is the activation of the chaotic unit and ω ($0 < \omega < 1$), α ($\alpha > 0$) and a are parameters of the Aihara model. The sigmoid function $f(y)$ is given by $f(y) = 1/(1 + e^{-y/\epsilon})$.

The time series generated by equation (22) is chaotic for certain sub ranges of the bifurcation parameter a in $[-1,1]$. This is demonstrated in the graphs in Figure 7 which show the cobweb iterations of this equation with two values of parameter a. The sigmoidal curve in each graph corresponds to the value of $x(t+1)$ for each value of $x(t)$ in the range $[-1,1]$. The cobweb diagram is then iterated by taking an initial value of $x(0)$, drawing a vertical line to the sigmoidal curve which gives the value of $x(1)$, then drawing a horizontal line to the diagonal, and then drawing a vertical line to the sigmoidal curve to give $x(2)$, and so on. In each of the graphs of Figure 7 the value of $x(0) = 0.2$. In graph (a), parameter $a = 0.74$, which gives rise a chaotic sequence of values. In graph (b), the value of $a = 0.5$ which results in a periodic sequence (in this case the sequence alternates between 0.333 and -0.333, the initial transients to the periodic sequence have been omitted from the graph).

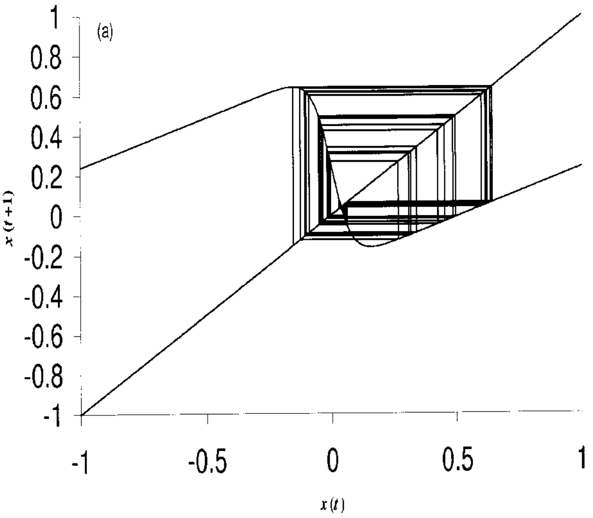
The network model presented here uses the value of $a = 0.74$, to ensure a chaotic firing sequence. An example chaotic time series for this parameter setting is shown in Figure 8. A strong positive average Lyapunov exponent ($\lambda = 0.295$) indicates that this is indeed a chaotic sequence.

The activation $y_i(t)$ of unit i on the chaotic layer is determined by the following equations:

$$g_i(t+1) = (1-\phi)(\omega y_i(t) - \alpha f(y_i(t)) + a) + \frac{\phi}{N_i}\sum_{j=1}^{N_i} y_j(t) \tag{23}$$

$$y_i(t) = g_i(t) + k_i(t) z_i(t) \tag{24}$$

The chaotic layer can be organised either linearly (Figure 9(a)), so that each unit has at most two lateral connections with its immediate neighbours, or it can be organized as a rectangular lattice (Figure 9(b)) so that each unit has at most four lateral connections. The right most term of equation (23) sums up the input from these lateral connections, with N_i denoting the number of neighbors for unit i. The constant ϕ determines the strength of the lateral connections relative to the units own chaotic



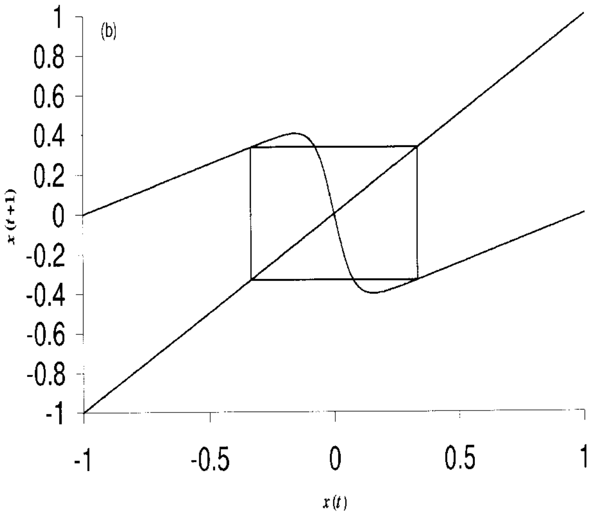


Figure 7. The cobweb iterations for the Aihara equation with (a) a = 0.74, (b) a = 0.5.

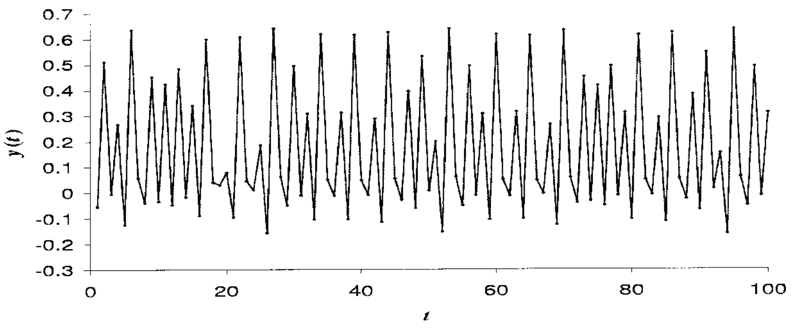


Figure 8. An example chaotic time series for the Aihara equation.

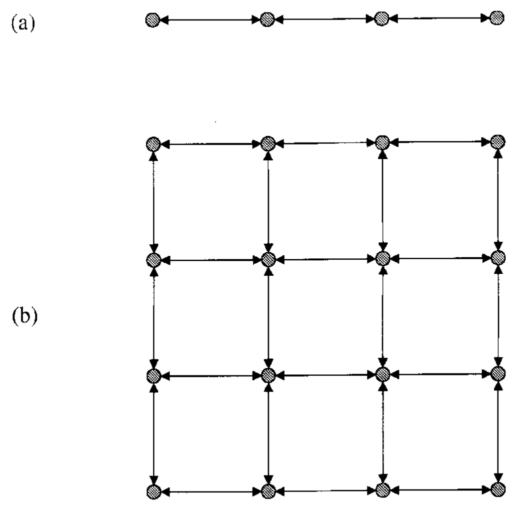


Figure 9. Two possible structures for the chaotic layer: (a) linear, (b) rectangular.

dynamics. The right most term of equation (24) introduces the control to be applied to this unit (see below). When this term is zero for a number of time steps, the dynamics of $y_i(t)$ are governed by the chaotic attractor of the Aihara equation.

Figure 10 shows the time series of a linear chaotic layer (Figure 9(a)) consisting of four units with no applied control. The average Lyapunov exponents of each unit is strongly positive, indicating that the dynamics of the connected units are chaotic ($\lambda_1 = 0.268734, \lambda_2 = 0.166363, \lambda_3 = 0.158992, \lambda_4 = 0.182272$).

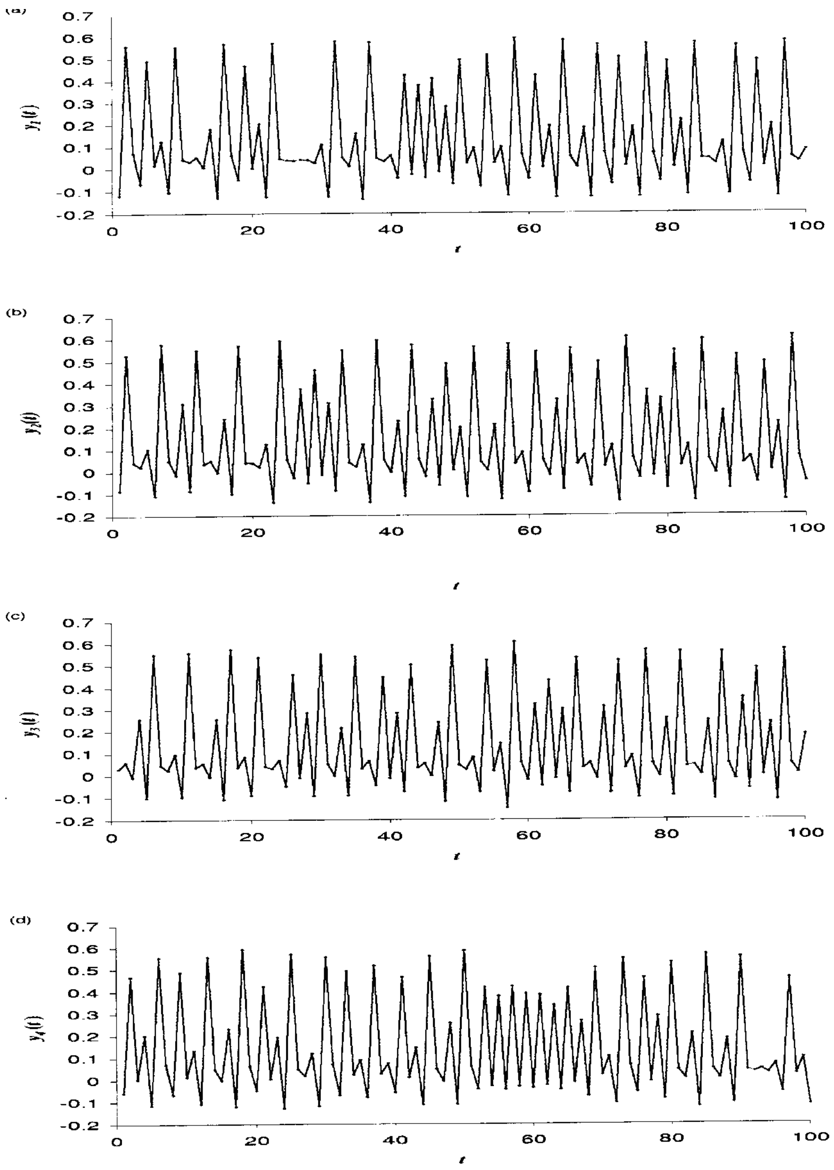


Figure 10. A chaotic time series generated by a 4-unit linearly connected chaotic layer.

Figure 11 shows the time series of four units taken from a 4×4 rectangular layer (Figure 9(b)) of chaotic units. Once again, the average Lyapunov exponents of each unit indicates that these time series are chaotic (see Table 1).

Each chaotic unit is associated with one unit in the inhibitory layer. The purpose of each inhibitory unit is to apply feedback control to stabilize the associated chaotic unit into a UPO. To achieve this, each inhibitory unit receives two inputs from the associated chaotic unit: an instantaneous input $g_i(t)$, and a multiple-delayed input:

$$\sum_{j=1}^{D} \xi_j y_i(t - j \times \langle \tau_i \rangle)$$

where τ_i is a *characteristic* time delay associated with inhibitory unit i ($\langle x \rangle$ denotes that x is rounded to the nearest integer value). We have found that a weighted sum of multiple delays is more effective at controlling discrete time equations than using a single delay value. The values of weight ξ_j decrease as j increases, so that more recent values in the evolution of $y_i(t)$ are given a higher weighting than older values. ξ_j is given by:

$$\xi_n = \frac{1}{n(1 + \sum_{k=1}^{D-1} \frac{1}{2^k})} \tag{25}$$

where D is the number of delays used. In our experiments, we found it sufficient to have $D = 3$.

The activation $z_i(t)$ of inhibitory unit i is given by:

$$z_i(t) = \begin{cases} 0 & : \quad \sum_{j=1}^{M} w_{ij}(t) I_j(t) = 0 \\ g_i(t) - \sum_{j=1}^{D} \xi_j y_i(t - j \times \langle \tau_i \rangle) & : \quad \sum_{j=1}^{M} w_{ij}(t) I_j(t) \neq 0 \end{cases} \tag{26}$$

where M is the number of input units, $w_{ij}(t)$ ($w_{ij}(t) > 0$) is the weight on the connections from input unit j to inhibitory unit i and $I_j(t)$ is the activation of the jth input unit. Note that the inhibitory unit's activation is gated by the presence of instantaneous input from the input layer: If there is no input to the network, the inhibitory units do not

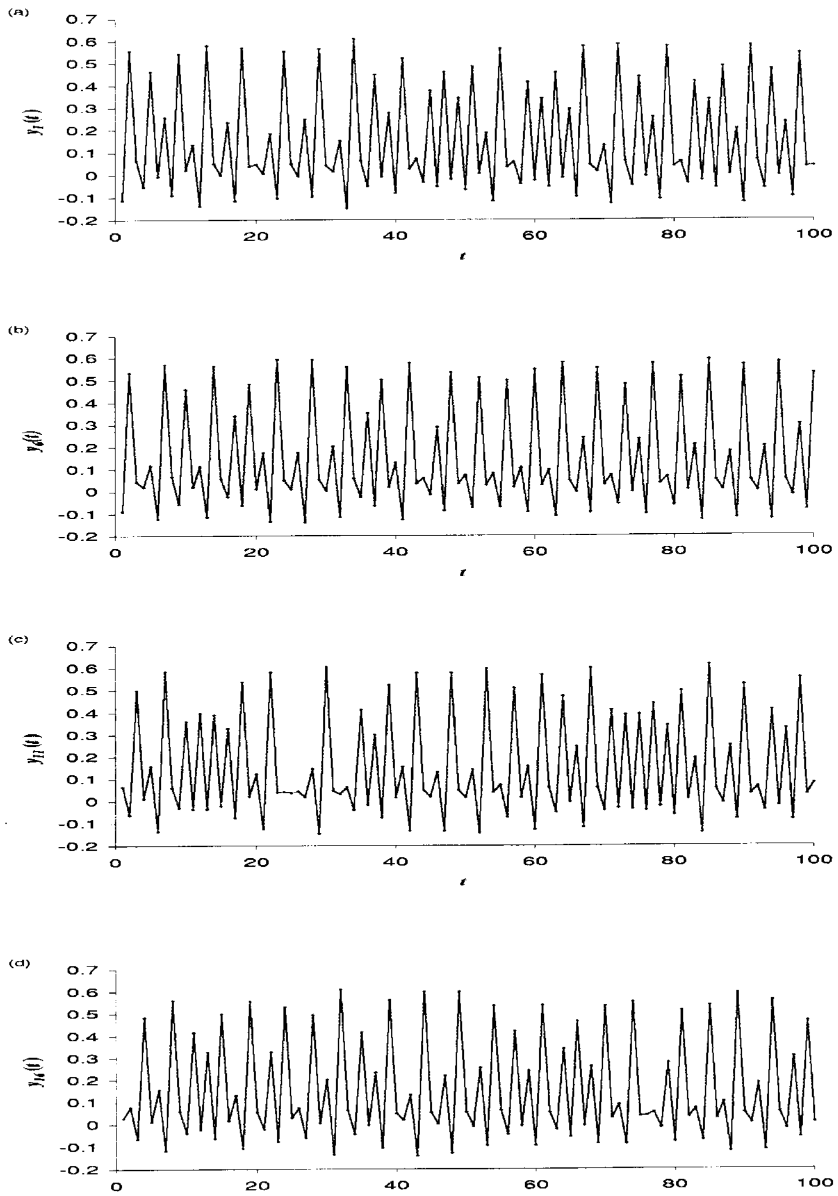


Figure 11. The chaotic time series generated by 4 units from a 16-unit rectangular chaotic layer.

Table 1. Average Lyapunov exponents for sixteen-unit rectangular layer

$\lambda_1 = 0.158137$	$\lambda_2 = 0.144228$	$\lambda_3 = 0.17781$	$\lambda_4 = 0.18886$
$\lambda_5 = 0.169635$	$\lambda_6 = 0.114216$	$\lambda_7 = 0.199601$	$\lambda_8 = 0.204769$
$\lambda_9 = 0.152558$	$\lambda_{10} = 0.0510107$	$\lambda_{11} = 0.20415$	$\lambda_{12} = 0.203094$
$\lambda_{13} = 0.194234$	$\lambda_{14} = 0.154669$	$\lambda_{15} = 0.0912817$	$\lambda_{16} = 0.0868024$

apply control to the chaotic units, thereby allowing them to follow their chaotic dynamics. The local coupling between the chaotic units ensures that a global chaotic dynamic *state* will emerge. The presence of this global chaotic state on the chaotic layer corresponds to a *non-recognition* state for the network. In other words, the network is not recognizing or classifying an input pattern when its chaotic layer is in a global chaotic state.

Figure 12 shows the time series of a four-unit linear chaotic layer. No control was applied during the first 50 iterations of this network, thereby allowing each unit to follow its chaotic dynamics. Control was then applied from $t = 51$ (i.e. $z_i(t) \neq 0$ for $t > 51$), with $\tau_i = 2.0$ for each inhibitory unit. The graph shows that each of the chaotic units quickly stabilizes to a period 2 orbit and remains within that orbit for as long as the control is applied. Figure 13 shows a similar experiment, but this time the control was applied with $\tau_i = 3.0$ for each inhibitory unit. In this case each chaotic unit stabilises to a period 6 orbit.

Figures 12 and 13 confirm that the length of the delay τ_i determines the period of the orbit which is subsequently stabilized. This fact is used in this model to relate input patterns to the orbits which are stabilized on the chaotic layer. This network is specifically designed to respond to the *dynamics* of the input signals it is presented with. Specifically, the network is sensitive to periodic elements which occur in those inputs. For example, an input might consist of the binary sequence (0, 1, 0, 1, ...), which contains a period 2 element. The association between the period of the input pattern and the delay applied for the control of the chaotic units is achieved by the connections from the input layer to the inhibitory layer. Each inhibitory unit has two connections to each of the units in the input layer: One is an instantaneous connection $I_j(t)$, the other is a delayed connection denoted by $I_j(t - \langle \tau_i \rangle)$, with τ_i being the characteristic time delay value for inhibitory unit i (each inhibitory unit uses the same time delay on all of its delayed input connections).

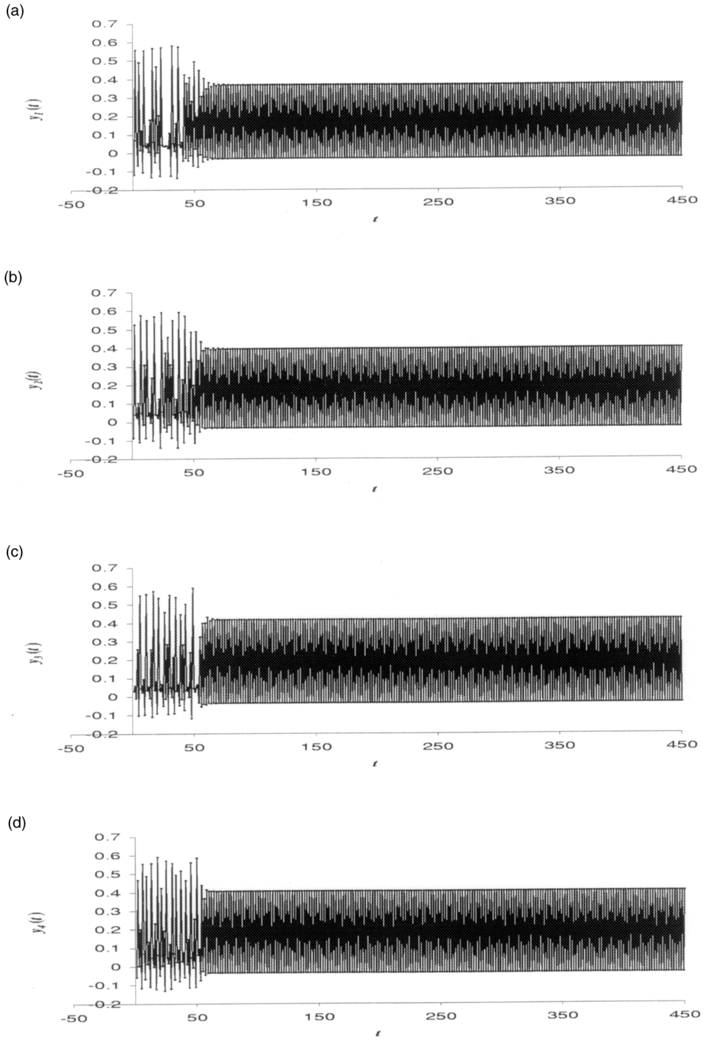
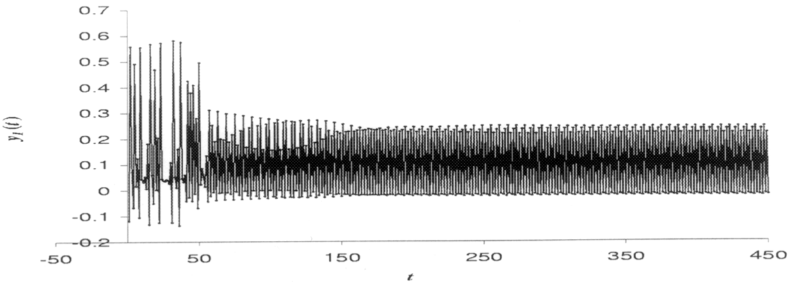


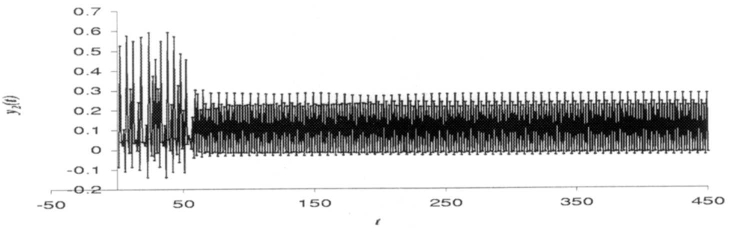
Figure 12. A 4-unit linear chaotic layer where each unit is stabilized to a period 2 orbit.

The characteristic delays for the inhibitory units are set to random real values at initialization. When an input signal is presented to the network, the inhibitory units are enabled and can apply control to the chaotic units. The network will select the unit in the inhibitory layer whose characteristic delay best matches the dominant period of the input

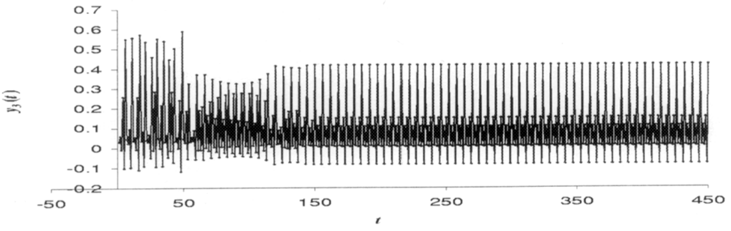
(a)



(b)



(c)



(d)

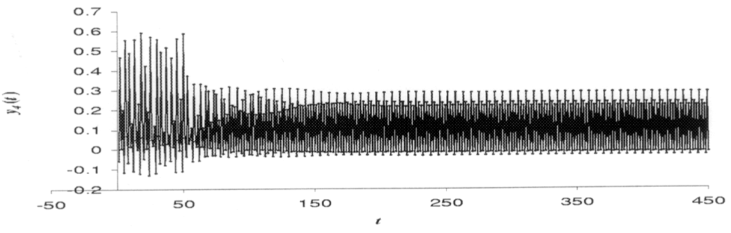


Figure 13. A 4-unit linear chaotic layer where each unit is stabilized to a period 3 orbit.

sequence. This selection of the *winning* unit is made by finding the unit which has the smallest value of $h_i(t)$:

$$h_i(t) = \sum_{j=1}^{M} w_{ij}(t)(I_j(t) - I_j(t - \langle \tau_i \rangle)) \tag{27}$$

The summation in this equation is close to 0 when the delayed activation $I_j(t - \langle \tau_i \rangle)$ is close in value to the instantaneous activation $I_j(t)$ on the input units whose weights $w_{ij}(t)$ are not close to 0. (Note that for each inhibitory unit $\sum_{j=1}^{M} w_{ij} = M$, so that not all of the input weights of an inhibitory unit can be close to 0 at the same time.) This means that $h_i(t)$ is close to 0 when the characteristic delay of inhibitory unit i is close to a frequency which is dominant in the input signal. In this way each inhibitory unit becomes a *feature detector* for the frequency profile of the input signal.

The Characteristic delay of the winning unit is denoted by τ_{win}. All inhibitory units i whose values of $\langle \tau_i \rangle$ are equal to $\langle \tau_{\text{win}} \rangle$ can apply control to their chaotic units. This is achieved through the following equation:

$$k_i(t) = \begin{cases} 0 & : \langle \tau_i \rangle \neq \langle \tau_{\text{win}} \rangle \\ \gamma & : \langle \tau_i \rangle = \langle \tau_{\text{win}} \rangle \end{cases} \tag{28}$$

where γ ($\gamma < 0$) is the optimum value that k_i can have for effective control to be applied to the chaotic unit.

ADAPTATION

The network architecture presented in Section 4 demonstrates how it is possible to generate internal dynamic recognition states (UPOs) which are associated with the dynamics present in the input signals. In this approach, memories are not stored as distributed patterns of weights between units, as is commonly used with artificial neural networks. Instead, *memory states* emerge from the dynamics of the network. Consequently, conventional approaches to network adaptation, which are centered on weight adaptation, cannot be applied in this model. In this section we present a novel approach to adaptation which is based on modifying network parameters in order to support the dynamics from which the memory states emerge.

The profile of characteristic delays across the inhibitory units determines which UPOs will be stabilized when input is presented to the network. Initially these delays are randomly selected. In which case neighboring inhibitory units can have characteristic delays which would attempt to stabilize different UPOs in their chaotic units. This would result in *local conflict* on the chaotic layer because of the lateral connections between neighboring units. Furthermore, it is possible that the input signals presented to the network may have a frequency profile which is not well fitted to the randomly selected characteristic delays of the inhibitory layer. Two competitive learning rules are introduced which enable the network to (i) adapt the weights on the connections from the input layer to the inhibitory layer so that input signals which are more commensurate with the characteristic delays of the inhibitory units can be given a stronger weighting, (ii) tune the characteristic delays to match the frequency profiles of the input signals, and (iii) develop a localized response on the inhibitory layer so that neighboring units have similar characteristic delays. In this way different regions of the inhibitory layer become feature detectors for certain characteristic frequencies of the input signals and dominant input frequencies would have a significant effect on the dynamics of the chaotic layer.

Both learning rules use the concept of a neighborhood around the winning unit which is commonly used in competitive learning neural networks (Kohonen 1989). The neighborhood is delimited by a radius $\mathcal{R}$ and a maximum reach $\mathcal{M}(t)$. Learning is applied to all units which are within a distance $\mathcal{M}(t)$ from the winning neuron. The distance d_{ij} from unit j to unit i in the inhibitory layer is defined as the minimum number of connections required to connect them. In Figure 14(b), for example, the distance $d_{1,49}$ be between unit 1 (top left of the diagram) and unit 49 (bottom right) is 12.

Figure 14 shows examples of the neighborhood radius $\mathcal{R}$ and maximum reach $\mathcal{M}(t)$ for a linear layer (a) and a rectangular layer (b). The neighborhood radius remains constant throughout training. All inhibitory units inside the neighborhood boundary (and inside the maximum reach if $\mathcal{M}(t) < R$) will have their characteristic delays and input weights adjusted so that they are better able to respond to the current input next time it is presented. Inhibitory units outside the neighborhood radius but within the maximum reach boundary (if

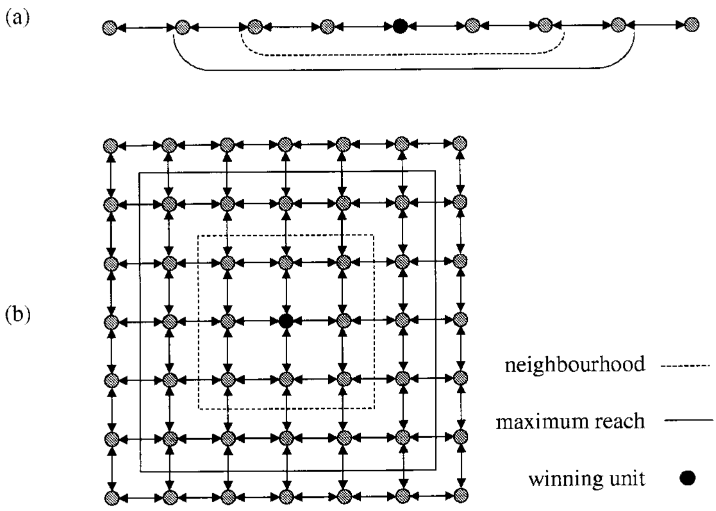


Figure 14. Examples of the neighborhood radius and maximum reach for (a) a linear chaotic layer, (b) a rectangular chaotic layer.

$\mathcal{M}(t) > R$.) will have their characteristic delays and input weights adapted so that they are less able to respond to the current input. In this way the regions of the inhibitory layer which respond to different frequencies in the input signals are pushed apart resulting in localized responses across the layer. The function $\rho(d_{ij})$ calculates the direction and magnitude of the changes to be made to the characteristic delay and input weights of an inhibitory unit i based on its distance d_{ij} from the winning unit j:

$$\rho(d_{ij}) = \frac{\eta(\mathcal{R} - d_{ij})}{\mathcal{R} * d_{ij}} \tag{29}$$

The maximum reach $\mathcal{M}(t)$ determines which units will be affected by the learning rules at time step t. To achieve the separation of units which respond to different frequencies in the input it is necessary for $\mathcal{M}(t)$ to start off larger than $\mathcal{R}$. However, if a unit repeatedly wins over several iterations then a large maximum reach would destroy previous adaptations on other units which were made in response to different frequencies on the input. To avoid this, when the winner changes from one unit to another in the inhibitory layer the maximum reach is set to a maximum value (which is constant) greater than $\mathcal{R}$. If the same units wins over several sequential iterations then $\mathcal{M}(t)$ is periodically decremented until it reaches zero, after which the only unit affected by the learning rules is the winning unit. If subsequently a different unit wins, then $\mathcal{M}(t)$ is set back to its maximum value and learning proceeds as before.

The first learning rule, LR1, is concerned with adapting the weights on the connections from the input layer to the inhibitory layer. These weights are important because the frequency profiles on each of the input units may differ significantly from each other, since each unit has its own independent input signal. This means that the characteristic delay of an inhibitory unit may only be commensurate with the frequencies of some of the input signals. Learning rule LR1 enables an inhibitory unit to shift weights away from input signals which are not commensurate with its characteristic delay, and towards units which are commensurate. LR1 is expressed by the following equation (note that for each inhibitory unit $\sum_{j=1}^{M} w_{ij} = M$):

$$m_{ij}(t) = |I_j(t) - I_j(t - \langle\tau_i\rangle)| \tag{30}$$

$$w_{ij}(t+1) = \begin{cases} w_{ij}(t) & : \; d_{ij} > \mathcal{M}(t) \\ w_{ij}(t) - \rho\left[\frac{w_{ij}(t)}{N} - 1 + \frac{w_{ij}(t)m_{ij}(t)}{\sum_{p=1}^{N} w_{ip}(t)m_{ip}(t)}\right] & : \; d_{ij} <= \mathcal{M}(t) \end{cases} \tag{31}$$

The second learning rule, LR2, is responsible for tuning the characteristic delays to match the frequency profiles of the input signals and developing a localized response on the inhibitory layer so that neighboring units have similar characteristic delays. This is achieved at each iteration by identifying the period τ_{ij} of the strongest input j to the winning unit i. The characteristic delay of the winning unit is then modified to be closer to the value of τ_{ij}. The modification of the characteristic delays of units within the maximum reach of the learning rule operates under the same conditions as LR1 with regard to the neighborhood radius.

The value of τ_{ij} can be calculated by storing a *pivot* value $p_{ij}(t) = w_{ij}(t)I_j(t)$ at time step t, and then comparing delayed inputs with the pivot value until $w_{ij}(t+\tau_{ij})I_j(t+\tau_{ij}) - p_{ij}(t) < \Delta$, where Δ is the tolerance level set for the comparison. When this condition is met,τ_{ij} can be taken as the period of that input signal at time $t + \tau_{ij}$. Each inhibitory unit uses this method to calculate the period of the signals on each of its instantaneous connections from the input layer. At each time step of the evolution of the system the winning inhibitory unit finds the period τ_{ij} of the input connection with the largest weight $w_{ij}(t)$. All units within the maximum reach of the learning rule then modify their characteristic delays according to the following equation:

$$\tau_k(t+1) = \begin{cases} \tau_k(t) & : \ d_{ij} > \mathcal{M}(t) \\ \tau_k(t) + \rho(d_{ij})(\tau_{ij}(t) - \tau_k(t)) & : \ d_{ij} <= \mathcal{M}(t) \end{cases} \tag{32}$$

where i is the index of the winning unit.

The following three experiments demonstrate how learning rules LR1 and LR2 adapt a four-unit linear layered network, and a 16-unit rectangular layered network. Input is presented to the network for $t > 50$ in all three experiments. In the first experiment a four-unit linear layered network was initialized to random characteristic delays for the inhibitory units. The network was then presented with two alternating input sequences.

In the fist sequence the input unit 1 was presented with a period 2 input (0, 1, 0, 1,...) lasting for 100 iterations, while input unit 2 was presented with a chaotic sequence over the same iterations. In the second sequence, input unit 1 was presenting with 100 values from a chaotic series whilst input unit 2 received a period three input (1, 0.5, 0, 1, 0.5, 0,...) over the same iterations. Since a chaotic sequence is by definition aperiodic, inhibitory units should shift their weights away from the chaotic input and towards the periodic input which is commensurate with their characteristic delay. The results of this experiment are presented graphically in Figures 15, 16, and 17.

Figure 15 shows how the values of the characteristic delays are modified by LR2 during learning. The final values of the characteristic delays are $\tau_1 = 2.42$, $\tau_2 = 2.72118$, $\tau_3 = 2.99575$ and $\tau_4 = 3$, showing that LR2 has been able to separate units responding to period 2 input (unit 1), from units responding to period 3 input (units 2, 3, and 4).

Figure 16 shows how LR1 enables each inhibitory unit to shift weights towards the input with the period which corresponds best to its characteristic delay, and away from the input which has a chaotic signal.

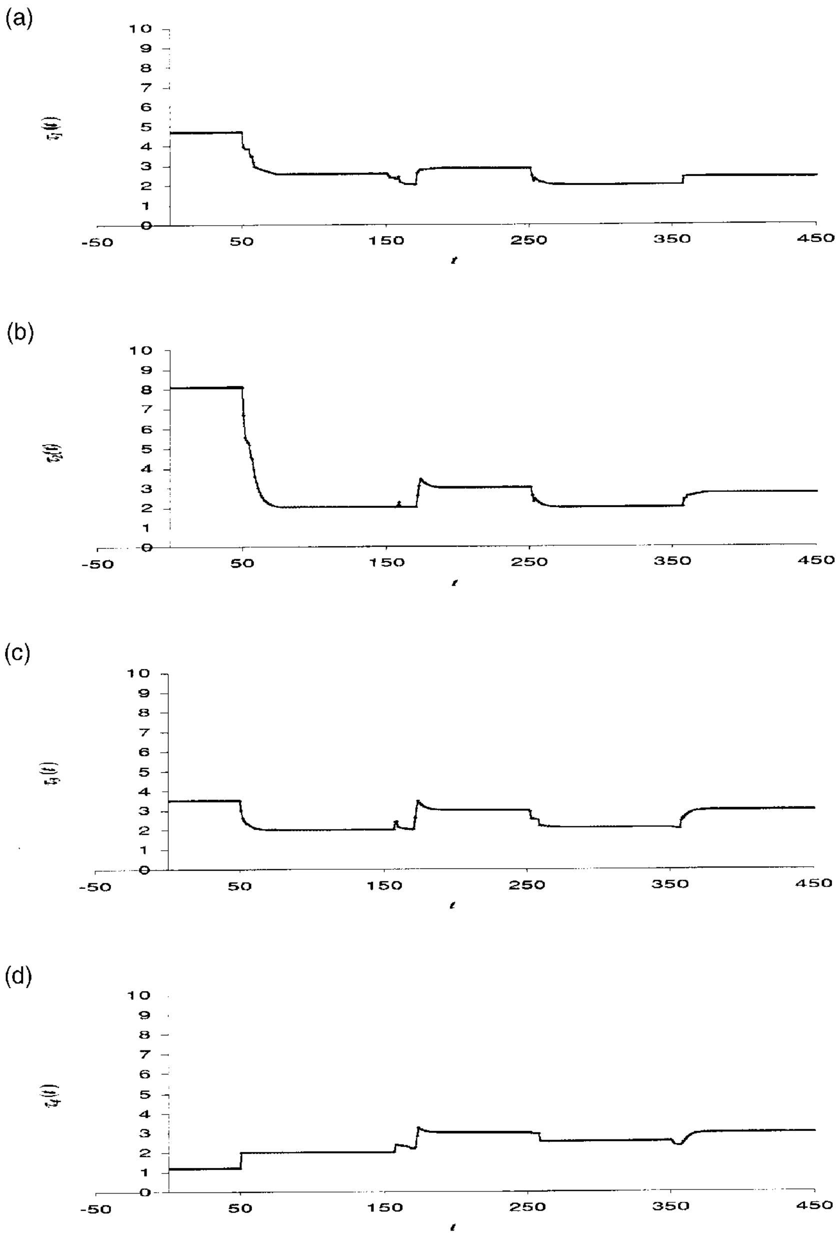


Figure 15. Adaptation of τ as a result of applying LR2 to a 4-unit linear network (experiment 1).

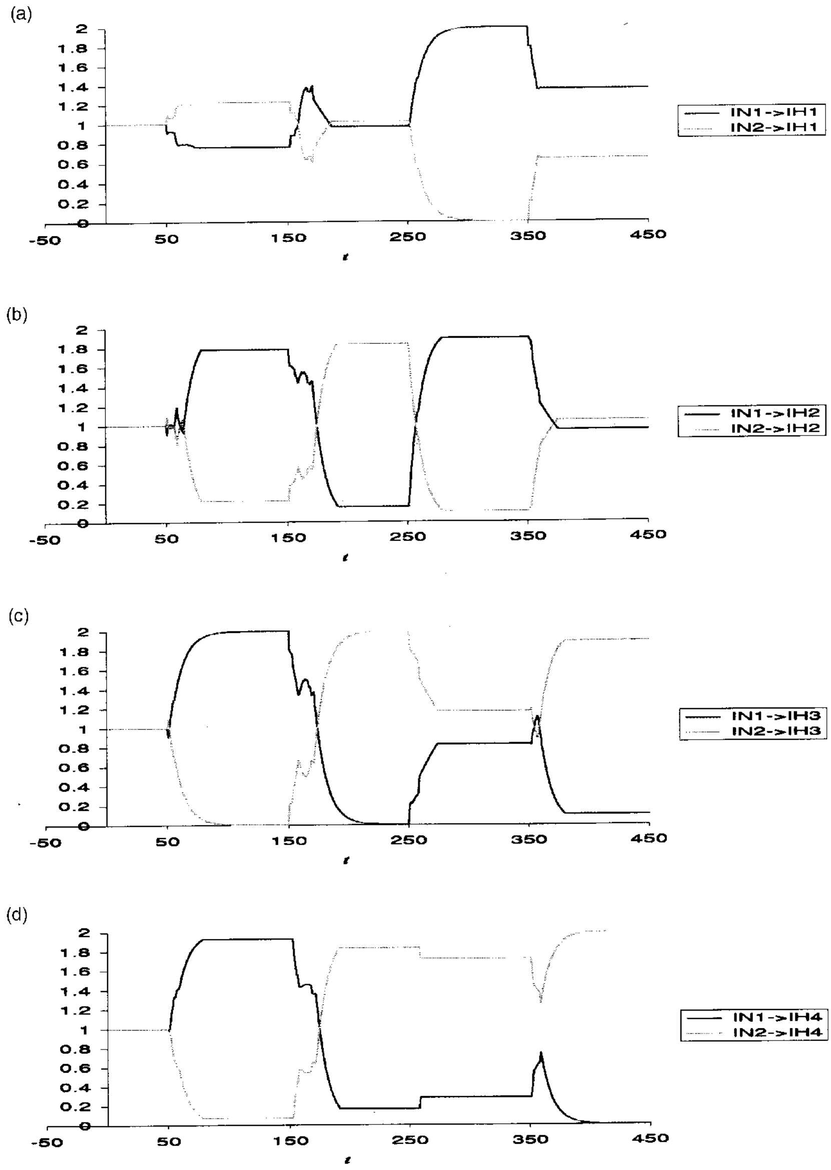


Figure 16. Adaptation of input weights which result for applying LR1 to a 4-unit linear network (experiment 1).

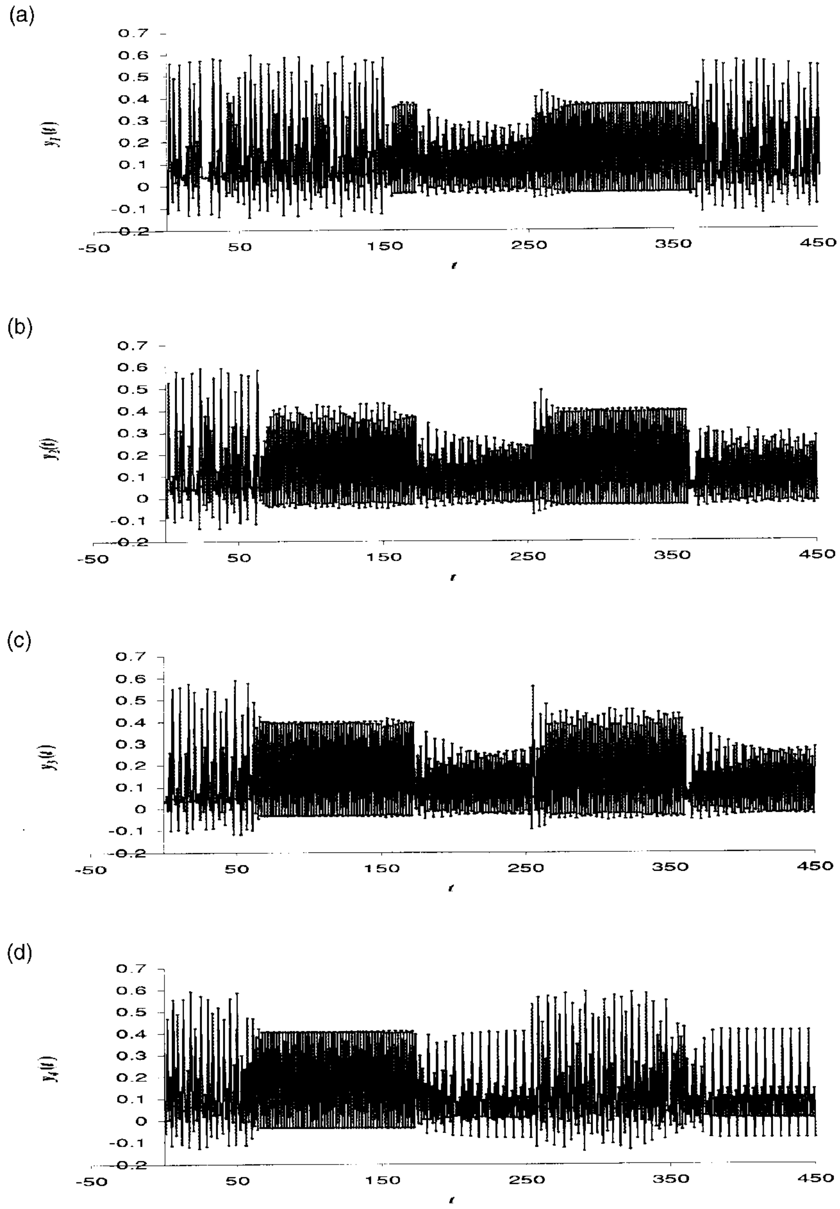


Figure 17. The activation time series of the 4-unit linear network during adaptation from LR1 and LR2 (experiment 1).

The final values of the weights are $w_{11} = 1.36046$, $w_{12} = 0.63954$, $w_{21} = 0.950279$, $w_{22} = 1.04972$, $w_{31} = 0.105788$, $w_{32} = 1.89421$, $w_{41} = 5.09E-05$, $w_{42} = 1.99995$. Note that unit 1 has shifted its weight towards input 1 where it finds the period 2 input is commensurate with the rounded value of its characteristic delay $\tau_1 = 2.42$. Similarly, units 2, 3, and 4 have shifted their weights towards the second input unit.

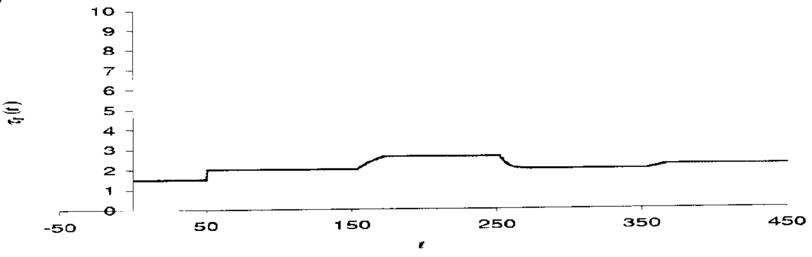
The graphs is Figure 17 show the activation of the chaotic units for this experiment. These results show that unit 1 learns to respond to period 2 input by stabilizing a period 2 orbit while that input is presented ($t \in [51..150, 251..350]$). Unit 4, on the other hand, learns to respond to a period 3 orbit by stabilizing a period 6 orbit when that input is presented ($t \in [151..250, 351..450]$). The units in between are compromised by their local connections and attempt to stabilize both orbits.

The second and third experiment involve a 4-unit linear network (Figure 9(a)), and the 16-unit rectangular network (Figure 9(b)). Both networks were presented with two input sequences: the first was period 2 (i.e. input unit I_1 was presented with the sequence (0, 1, 0, 1,...), and input unit I_2 was presented with the sequence (1, 0, 1, 0,...)), the other was period 3 (i.e. I_1 was presented with (0, 0.5, 1, 0, 0.5, 1,...) and I_2 with (1, 0.5, 0, 1, 0.5, 0,...)). In each case the input was started at $t = 51$, and consisted of alternating 100 of the period 2 iterations with 100 of the period 3 iterations. The results of applying adaptation to the characteristic delays for the 4-unit and the 16-unit network are shown in Figures 18 and 20, respectively. The activations of the chaotic units are plotted in Figures 19 and 21.

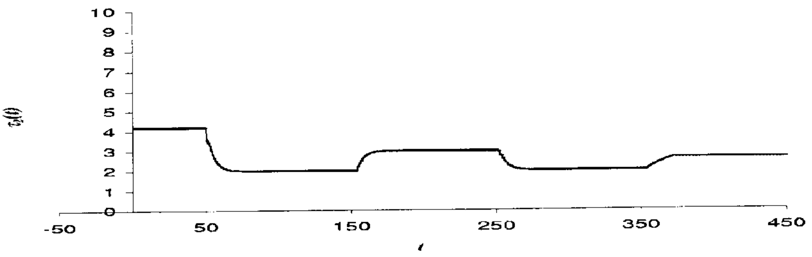
The final values of the characteristic delays for the 4-unit linear network are $\tau_1 = 2.19627$, $\tau_1 = 2.60376$, $\tau_1 = 2.99479$, $\tau_1 = 3$. Note that unit 1 is a feature detector for inputs with period 2, while units 2, 3, and 4 have become feature detectors for inputs of period 3. Consequently, Figure 19 shows that chaotic unit 1 quickly stabilizes a period 2 orbit in response to a period 2 input (for $t \in [51..150, 251..350]$). Unit 4 on the other hand eventually stabilizes for period three input (for $t \in [151..250, 351..450]$). The units in between are affected by both periods and so have a mixed response.

The final values of the characteristic delays for the 16-unit rectangular network are shown in Figure 22. This figure clearly shows that the inhibitory layer has been partitioned into units which have $\langle\tau_i\rangle = 2$ and units which have $\langle\tau_i\rangle = 3$. Figure 21 shows the activation of two units from each partition. Units 4 and 7 have $\langle\tau_i\rangle = 3$, and so develop a

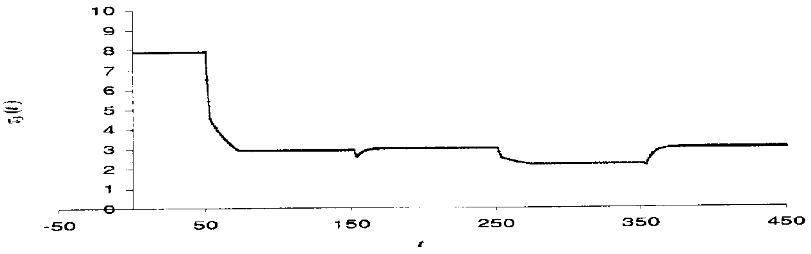
(a)



(b)



(c)



(d)

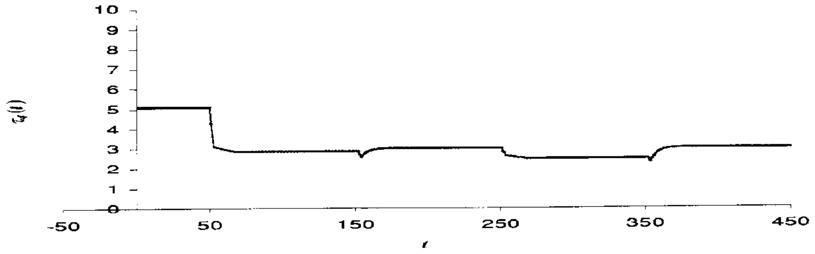
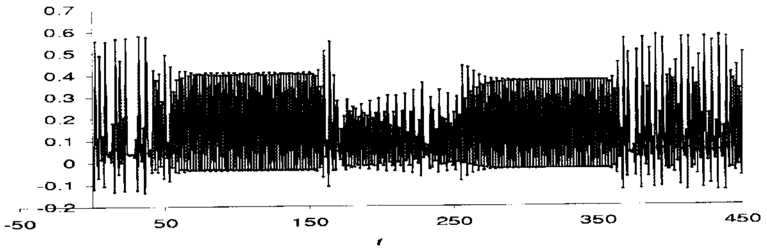
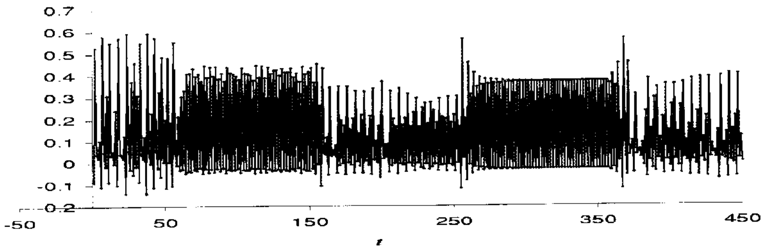


Figure 18. The characteristic delays of a 4-unit linear network presented with period 2 and 3 input sequences (experiment 2).

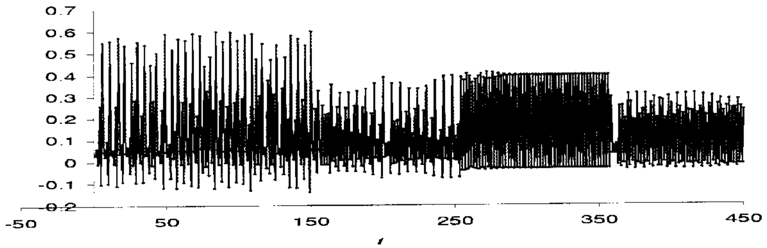
(a)



(b)



(c)



(d)

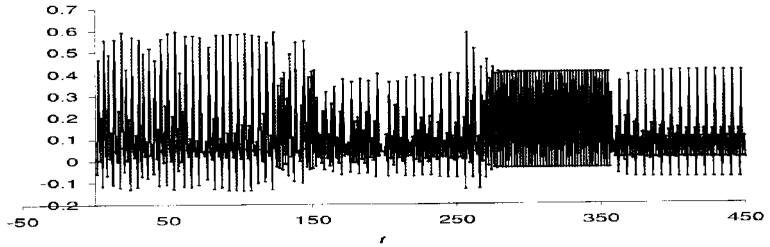


Figure 19. The activation time series of a 4-unit linear network presented with period 2 and 3 input sequences (experiment 2).

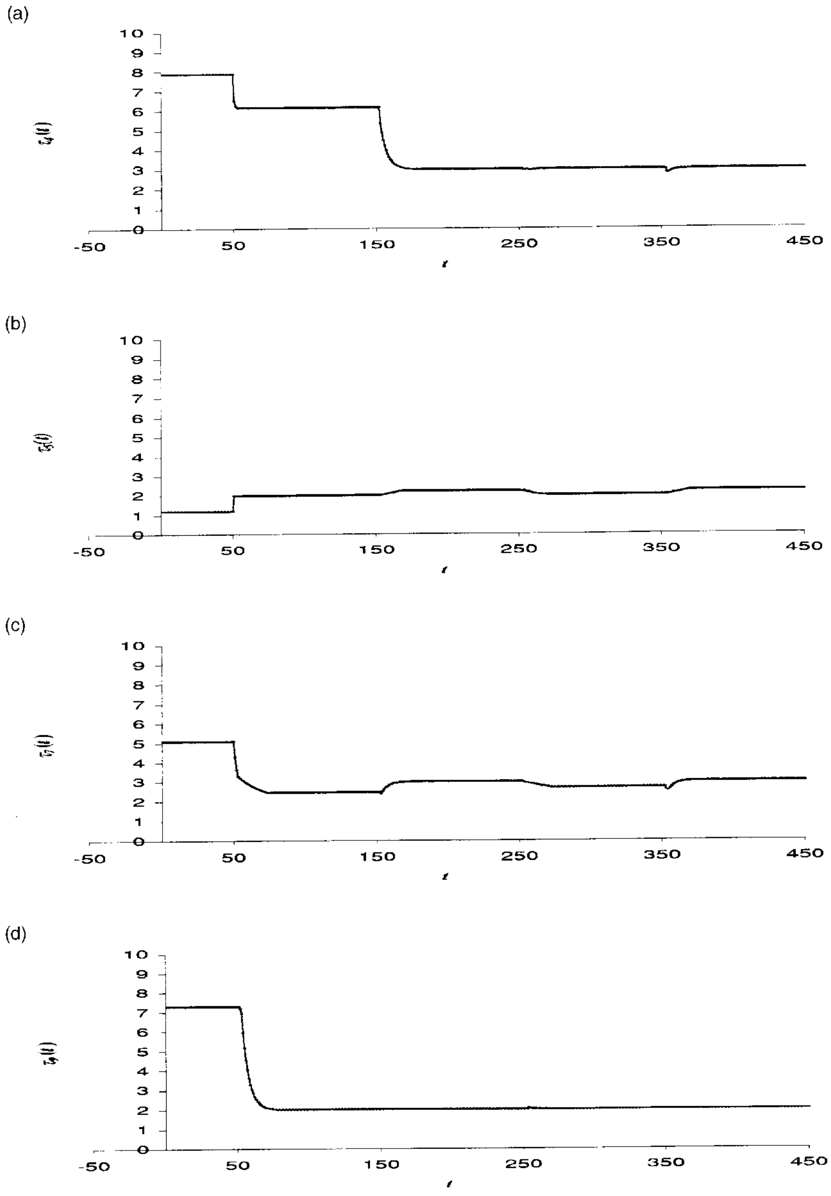


Figure 20. The characteristic delays of 4 units taken from a 16-unit rectangular network presented with period 2 and 3 input sequences (experiment 3).

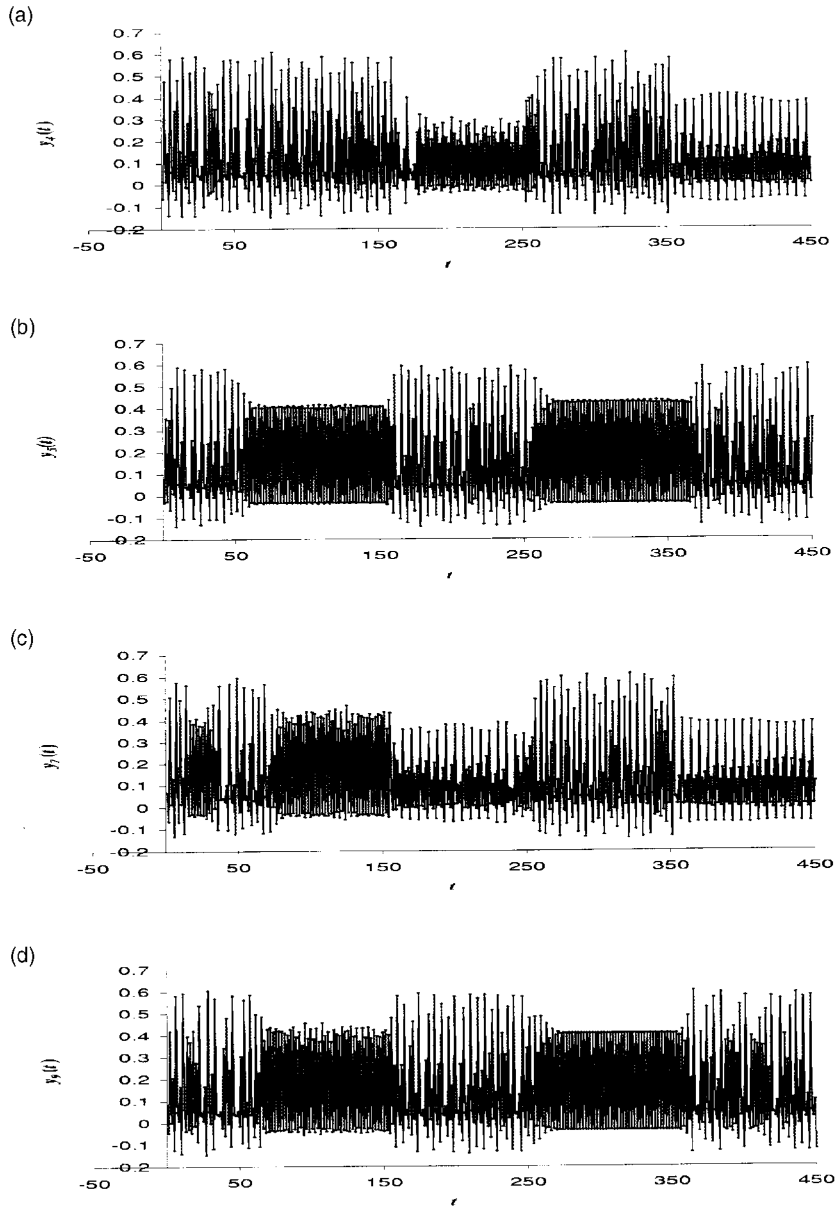


Figure 21. The activation time series of 4 units taken from a 16-unit rectangular network presented with period 2 and 3 input sequences (experiment 3).

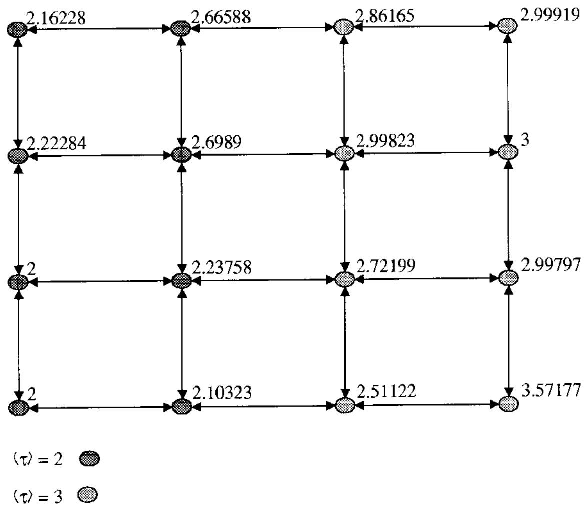


Figure 22. The final values of the characteristic delays for a 16-unit rectangular network trained on period 2 and period 3 inputs (experiment 3).

response to period 3 input by stabilizing a period 6 orbit. Units 5 and 9, on the other hand, have adapted their characteristic delays so that $\langle \tau_i \rangle = 2$, enabling them to respond to a period 2 input.

DISCUSSION

The results presented in this paper demonstrate the ability of a discrete neural network to respond on periodic input by stabilizing into an unstable periodic orbit. Delayed feedback control is used to respond to the period of the input and the behavior of the network is affected accordingly. Varying the input patterns, period-2, period-3 or chaos allows the system to distinguish between information and the absence thereof.

The spatial organization of the network makes it possible to allow subsets of neurons to stabilize into different orbits depending on the presented input. By dynamically modifying the characteristic delays of the inhibitory units, the system can be trained to stabilize appropriate orbits. In addition the input weights may be modified to stabilize the sensitivity of a subset of neurons for a particular input frequency.

The activations on the chaotic layer act as a kind of dynamic representation. This representation consists of temporally and spacially distributed patterns of activity across the chaotic units. When input signals are present, the network has to resolve the reduced dynamics from the input patterns with the chaotic behavior. Both of these types of dynamics are responsible for generating the stabilized dynamic patterns of part of the network which are the internal recognition state for the input signals being presented. One of the appealing aspects of this model is that the self-organized representations which emerge are a consequence of the resolution of internal and external dynamic states.

A significant aspect of this model is that the representations (or memories) generated are not stored as distributed patterns of weights across network connections. Rather, the representations are embedded in the dynamics of the system. This raises questions about the role of learning and weight adaptation in networks like this. The adaptation rule for the weights from the input layer to the inhibitory layer (LR1) is based on extracting *dynamic* features of periodic input sequences. The weight $k_i(t)$ is modified according to the fit of the characteristic delay of the inhibitory unit against the frequency profile of the input unit. Both of these types of weight changes are aimed at supporting the *dynamics* of the network from which the representations emerge. The second learning rule (LR2) is also designed to support the dynamics of the network. This learning rule modifies the characteristic delays of the inhibitory units, which in turn determine which orbits are stabilized on the chaotic layer. This is done by selecting delays that are inherent in the frequency profile of the input signals. The application of both learning rules ensures that the network is able to differentiate input sequences which have different frequency profiles. In other words, the network operates as a pattern classifier.

Further work on this approach will necessarily involve (i) assessing how well this network model is able to generalize when presented with noisy input and (ii) quantifying the capacity of the network in terms of the number of distinct classes of input patterns it is able to recognize without serious loss of performance. Meanwhile, the results presented here demonstrate that it is possible to use chaos as a method of pattern recognition, where recognition states are dynamic orbits which have been stabilized according to the frequency profiles of the input sequences. We have also addressed the issue of adaptation in the context of a dynamic network of this kind.

REFERENCES

Aihara, K., T. Takabe, and M. Toyoda. 1990. Chaotic neural networks. *Physics Letters A* 144(6,7):333–339.

Andreyev, Y. V., Y. L. Belsky, A. S. Dmitriev, and D. A. Kuminov. 1996. Information processing using dynamical chaos: neural networks implementation. *IEEE Transactions on Neural Networks* 7(2):290–299.

Chapeau-Blondeau, F., and G. Chauvet. 1992. Stable, oscillatory and chaotic regimes in the dynamics of small neural networks with delay, *Neural Networks* 5:735–743.

Crook, N. T., C. H. Dobbyn, and T. V. S. M. olde Scheper. 2000. Chaos as a desirable stable state of artificial neural networks. *Advances in Soft Computing: Soft Computing Techniques and Applications*. R. John and R. Birkenhead, eds. Heidelberg: Physica-Verlag, 52–60.

Destexhe, A. 1994. Oscillations, complex spatiotemporal behavior and information transport in networks of excitatory and inhibitory neurons. *Physical Review E* 50(2):1594–1606.

Freeman, W. J. 1985. Strange attractors in the olfactory system of rabbits. *Electroencephalography and Clinical Neurophysiology* 61:S155–S155.

Freeman, W. J. 1987. Simulation of chaotic EEG patterns with a dynamic model of the olfactory system. *Biological Cybernetics* 56:139–150.

Freeman, W. J., and J. M. Barrie. 1994. Chaotic oscillations and the genesis of meaning in cerebral cortex. *Temporal Coding in the Brain*. G. Buzsaki, R. Llinas, W. Singer, A. Berthoz, and Y. Christen, eds. Berlin: Springer-Verlag, 13–37.

Hoshino, O., N. Usuba, Y. Kashimori, and T. Kambara. 1997. Role of itinerancy among attractors as dynamical map in distributed coding scheme. *Neural Networks* 10(8):1375–1390.

Kaneko, K., and I. Tsuda. 1994. Constructive complexity and artificial reality: an introduction. *Physica D* 75:1–10.

Klotz, A., and K. Bräuer. 1999. A small-size neural network for computing with strange attractors. *Neural Networks* 12:601–607.

Kohonen, T. 1989. *Self-organization and Associative Memory*. (3rd ed), Berlin: Springer-Verlag.

Kushibe, M., Y. Liu, and J. Othsubo. 1996. Associative memory with spatiotemporal chaos control. *Physical Review E* 53(5):4502–4508.

olde Scheper, T. 2001. Chaos and Information in dynamic neural networks. Ph.D. Thesis, Oxford Brookes University, Oxford, UK.

Ott, E. 1993. *Chaos in Dynamical Systems*. Cambridge, UK, Cambridge University Press.

Ott, E., C. Grebogi, and J. A. Yorke. 1990. Controlling chaos. *Physical Review Letters* 64(11):1196–1199.

Pasemann, F., and N. Stollenwerk. 1998. Attractor switching by neural control of chaotic neurodynamics, *Network: Computational Neural Systems* 9:549–561.

Pyragas, K. 1992. Continuous control of chaos by self-controlling feedback. *Physics Letters A* 170.

Pyragas, K. 1995. Control of chaos via extended delay feedback. *Physics Letters A* 206:323–330.

Rössler, O. E. 1976. An equation for continuous chaos. *Physics Letters* 57A(5):397–398.

Schuster, H. G., ed. 1999. *Handbook of Chaos Control.* Weinheim, Germany, Wiley-VCH Verlag GmbH.

Sinha, S., and W. L. Ditto. 1999. Computing with distributed chaos. *Physical Review E* 60(1):363–377.

Tsuda, I. 1996. A new type of self organization associated with chaotic dynamics in neural networks. *International Journal of Neural Systems* 7:451–459.

Tsui, A. P. M., and A. J. Jones. 1999. Periodic response to external stimulation of a chao in neural networks with delayed feedback. *International Journal of Bifurcation and Chaos* 9(4):713–722.

Watanabe, M., and K. Aihara. 1997. Chaos in neural networks composed of coincidence detector neurons. *Neural Networks* 10(8):1252–1259.

Watanabe, M., K. Aihara, and S. Kondo. 1998. A dynamic neural network with temporal coding and functional connectivity. *Biological Cybernetics* 78:87–93.