AIAA Meeting Papers on Disc, January 1997
A9715709, AIAA Paper 97-0686

Connectionist algorithms for identification and control - System structure

David C. Hyland
Michigan Univ., Ann Arbor

AIAA, Aerospace Sciences Meeting & Exhibit, 35th, Reno, NV, Jan. 6-9, 1997

This paper gives a variety of theoretical results associated with the Adaptive Neural Control (ANC) architecture and its application to the control of flexible structures. ANC is a new parallel processing, decentralized architecture for identification and adaptive control that has been under development by the author and associates over the past five years. The ANC architecture consists of a hierarchy of standardized modules and rules for combining them. In this paper, we give a step-by-step description of the ANC components and present basic theorems on convergence and stability for applications involving both identification and adaptive control. (Author)

AIAA-97-0686

CONNECTIONIST ALGORITHMS FOR IDENTIFICATION AND CONTROL: SYSTEM STRUCTURE AND CONVERGENCE ANALYSIS

David C. Hyland
Department of Aerospace Engineering
The University of Michigan, Ann Arbor, MI

Abstract

This paper gives a variety of theoretical results associated with the Adaptive Neural Control (ANC) architecture and its application to the control of flexible structures.

ANC is a new parallel processing, decentralized architecture for identification and adaptive control that has been under development by the author and associates over the past five years. The ANC architecture consists of a hierarchy of standardized modules and rules for combining them. In this paper, we give a step-by-step description of the ANC components and present basic theorems on convergence and stability for applications involving both identification and adaptive control.

Introduction

Despite recent advances in efficiency, current methodologies for space system control still engage significant human resources for engineering development and maintenance. For example, since fixed-gain space system controllers must be updated periodically to adjust to in-mission changes in system dynamics, this implies burdensome ground support activities. The development of the Adaptive Neural Control (ANC) architecture by the author and associates over the past several years is part of an effort to develop neural network based controllers capable of self-optimization, on-line adaptation and autonomous fault detection and control recovery. This development also supports the Nation's long-term space exploration objectives for which antonomous spacecraft involving self-reliant control systems are a necessity.

The ANC architecture is a set of building blocks and rules for combining them so as to achieve massively parallel and decentralized processing for identification and adaptive control of dynamic systems. A sequence of papers[1-7] have outlined the ANC scheme and reported numerous simulation and experimental results. While these previous papers display important accomplishments in realizing neural control systems in hardware, the present paper gives a step-by-step description of the ANC components and presents basic results on convergence, and stability.

First, we describe the more or less direct antecedents of ANC in the literature of 'connectionist' or 'artificial neural network' systems. Of course, the foundation of this work is the basic neuron model of McCulloch and Pitts[8], the early decentralized adaptation algorithm of Hebb[9] and the basic multi-layer feedforward structures defined by Rosenblatt[10-13]. Given this basis, the ANC approach utilizes gradient based learning schemes. Subsequent to the advances of Widrow and Hoff[14-16] wherein the gradient of the instantaneous error is used, the more general backpropagation algorithm was invented by Werbos[17] and independently developed by Parker[18], LeCun[19], Rumelhart, Hinton and Williams[20,21] and Rumelhart and McClelland[22]. It will be seen below that the ANC architecture involves a form of local backpropagation. There are various approachs to the application of backpropagation systems to temporal processing (of time series signals). These may be epitomized by the time-delay neural networks of Lang and Hinton[23] or Waibel et al.[24] and the finite-impulse response (FIR) multi-layer perceptrons of Wan[25]. ANC is more closely related to the latter approach. Associated with this scheme are several basic training algorithm approaches: "backpropagation through time" as explained by Werbos[17], Rumelhart et al[22] and Werbos[26], the "temporal backprop" of Wan[27,28] and the "real-time temporal learning" approach of Williams and Zipser[29]. The latter approach, to which ANC bears the closest resemblance, combines multilayer perceptrons with tapped delay lines and trains the network using the "temporal learning algorithm". This algorithm requires the use of auxiliary "sensitivity systems" to compute gradients. This scheme was applied by Narendra and Parthasarathy[30] to treat dynamic system identification and control. In contrast, the ANC approach decentralizes computations associated with both the forward signals and the backpropagated signals and adaptation occurs locally in each synapse. Because of the "local backpropagation" arrangement, no separate sensitivity systems are required.

In addition, as is discussed below, there are further novel features of ANC. One of the most important of these is the time varying rule for

assigning the adaptive speed coefficient. In most neural network schemes, the adaptive speed is either a constant or is made to be time-varying in a more or less ad-hoc way as to speed up convergence. The time-varying adaptive speed utilized in ANC greatly simplifies the investigation of convergence and allows a guarantee of convergence under certain broad conditions.

In the following section, we describe the hierarchy of ANC components and how they are used to build up more complex systems.

ANC Architecture Description

The processing architecture described here combines modular, standardized components to achieve its objectives. Modularity permits the parameters of the controller device to be fine tuned to specific applications with minimum development effort. The massive parallelism and decentralization of the architecture allows great latitude in the allocation of the computational burden and implies considerable implementation flexibility.

The hierarchy of modular structures that compose the ANC architecture is depicted in Figure 1. In the order progressing from basic constituents to higher-level modules, the modular structures include individual neurons which are interconnected by synapses and dynamic arrays which are in turn, grouped into replicator networks, one or more of which may form an adaptive neural control system. For ease of explanation, all elements in the hierarchy are assumed to operate within the framework of a *sampled data* computational system — i.e. all external inputs are sampled periodically at some particular sample rate, all devices behave as discrete time systems and all outputs to external devices (e.g. actuators for control) are put through digital-to-analog converters. Thus, inputs and outputs as well as internal signals will be regarded as discrete time series with the time value denoted by some integer value.

The lowest level of the hierarchy as shown in Figure 1 contains three devices: an individual *neuron*, an individual *synaptic connector* and a tapped delay line.

For present purposes, a tapped delay line (TDL) is a device that takes a scalar time series input and produces an L-dimensional vector output consisting of the present value of the input and its L-1 delayed values. The result of passing any signal $(\cdot)$ through a TDL is denoted by an over bar: $\bar{(\cdot)}$. At the bottom of the hierarchy, the basic neuron can be defined as in Figure 2. Unlike the one-way neurons usually postulated in neural net applications to adaptive control. (e.g. Williams and Zipser[29], Narendra[30]) the neuron defined here is inherently a two-way device with a *forward* signal flow path and a *backward* signal flow path. The neuron also receives a forward path bias input and a backward path input. Signals along the forward path are operated on by the neural function, which may be a sigmoid function as illustrated in Figure 2a. Signals along the backward path are multiplied by the derivative of the neural function. The forward and backward paths receive input signals from other neurons via synaptic connectors. On the forward path, during each computational cycle, the neuron sums the forward path inputs from the synaptic connectors and adds a forward path bias signal to form signal u_k (see Figure 2a). The bias signals are used to connect the system with the outside world, e.g. the forward bias might be a training stimulus injected to drive the learning process in the overall network. Signal u_k is then passed through the neural function (signoid nonlinearity), $g(x)$, (called the *activation function*) to form the forward path output of the neuron, x_k. The neural function $g(x)$, may be assumed to be differentiable and to have an approximately linear range for small values of x and to saturate at large values, approaching +1 as $x \rightarrow \infty$ and -1 as $x \rightarrow \infty$. The range of the values of the argument over which g is linear is a variable specification. In particular, as a special case, $g(x)$ can be linear; i.e. $g(x)=x$. In its forward path operation, the neuron is essentially the same as the customary definitions.

Fully localized computational capability is provided by the backward path operations that are executed simultaneously with the forward path. The neuron sums the backward path bias inputs (which may represent error signals injected into the network) and the backward path inputs. This signal is then multiplied by $g'(u_k)$ to form the backward path output signal y_k. $g'(\cdot)$ is the function formed from the derivative of $g(\cdot)$, and because of the characteristics assumed for $g(\cdot)$, is nonnegative with a single maxima at zero. Thus, the neuron has a pair of inputs and a pair of outputs, each member of the pair being associated with signals flowing in opposite directions.

With reference to Figure 2b, the neuron may be depicted in a simplified block diagram convention. The neuron as a whole is represented by a hexagon. Only the forward path signals are shown explicitly (the backward path operations can be inferred from the forward path). Forward path bias signals are always shown entering the top of the hexagon symbol and the backward path biases are shown entering the bottom.

The characteristics (such as $g(\cdot)$ and $g'(\cdot)$) of the neuron in a particular system are fixed and are not adapted over time. The capability to adapt over the time actually resides, not in the neuron, but in the *synaptic connectors* (or simply synapses) which are elements that serve to link several neurons together. Referring to Figure 3, a synaptic connector is also an inherently two way device with both forward and backward inputs and outputs. On a forward signal path the synapse merely multiplies the input x by a real number, $W(n)$ to produce an output y. $W(n)$ is the *synaptic weight* or, more simply, the "weight" of the connector (sometimes called a "gain"). On a backward path, the same kind of operations occur: $\bar{x}$ is multiplied by $W(n)$ (the same weights as the forward path) to produce the backward path output
. Essentially the synapse is analogous to a bi-directional cable with the same resistance in both directions. In addition the synapse may also adjust its own weight. For example the synapse may use the formula in Figure 3. The change in the weight from time n to time $n+1$ may be simply proportional to the product of the forward path input to the synapse and the backward path input to the synapse. This update rule, is purely decentralized or localized since it uses only information available to the individual synapse. Of course, for each synaptic connector there is the option of *constraining* the weight — to a constant or equal to some externally supplied time varying signal.

The constant of proportionality in Figure 3, $\mu(n)$, is called here the *adaptive speed* since it governs the rapidity of learning. As discussed below, the *adaptive speed* is not necessarily a constant and may be updated (using local information) so as to guarantee convergence in both system identification and adaptive control.

With reference to Figure 3b the simple block diagram convention for the synapse is a directed line with the synaptic weight indicated above the arrow. Once again, the backward path operations are implied while only the forward path signals are shown explicitly.

Neural systems can be constructed consisting of several neurons linked by the synaptic connectors. To clarify how these bi-directional devices fit together, a three neuron, two synapse system has been constructed in figure 4a. Figure 4b depicts the same arrangement of neurons and synapses as in Figure 4a using the simplified notation identified in figures 2b and 3b. This serves to illustrate how the two-way devices 'dove-tail' and how the simple block diagrams are to be interpreted.

Systems built up of the neurons and synapses defined above are adequate to address *static* pattern classification and nonlinear mapping problems. By virtue of the backward signal path defined at the most fundamental level, backpropagation of error and adaptive learning capabilities are built in. Moreover, the learning capability is totally localized and decentralized — separate "sensitivity systems" and weight update computations are not needed. Each subdivision of a neural network composed of the elements defined above has a localized learning capability. This feature confers a high degree of fault tolerance — i.e. with a completely decentralized learning architecture, failure or damage to a subset of neurons will result in the remaining neurons eventually assuming the functions of the damaged components.

However, there remains the problem of applying networks of this type to tasks involving dynamic systems with time-varying input and output signals. This problem is addressed by the second member of the hierarchy shown in Figure 1. The key to applying the neurons and synapses of Figures 2a and 3a to dynamic system identification is to organize the neurons into larger building blocks — termed the dynamic arrays. As illustrated in Figure 5a arrays may be connected by *Toeplitz synapses*. An array is a "stack" of neurons such that neurons within the same stack are not connected, but are (at most) only connected to neurons in some other stack. Basically, this organization into stacks introduces a temporal ordering into the network: the position of a neuron in the array indicates the "age" of the data (relative to the present instant) it is meant to process. A neuron that is further from the top of the stack represents a time instant that is further into the past.

In general, two such stacks of neurons can be interconnected with any number of connections. However, it is desirable to impose connectivity constraints such that the neuron k places from the top in one array receives (forward path) signals only from the neurons in another stack that are *k places or more* from the top. Any bundle of synaptic connectors obeying this constraint is termed a *Toeplitz synapse*. As illustrated in Figure 5a, the weights of this bundle can be represented as an upper triangular matrix. Fundamentally, this upper triangular structure is designed to preserve temporal ordering and causality within higher order networks. In other words, by forbidding a top level neuron from one stack from feeding signals into a lower level neuron in another stack, the system is constrained to be capable of modelling only *causal* systems in which the future

input signals cannot influence the past output signals.

As indicated in Figure 5b the arrays and synaptic connections may be simply diagrammed as shown. As indicated, arrays are represented by double-lined hexagon symbols. Toeplitz synapses are indicated by double arrows with the associated weight matrix shown within the arrow. If two or more Toeplitz synapse arrows converge on the imput to an array, the input is the sum of the indicated symbols. As before, only the forward path signal flows are shown explicitly.

Any network formed by an arbitrary number of arrays, linked by Toeplitz synapses in any arbitrary pattern (series connections, parallel, closed-loops or combinations thereof) is referred to herein as a *Teoplitz network*. The two top levels of the hierarchy in figure 1 are examples of Toeplitz networks, assuming Toeplitz synapses are being used. Considerable details on the construction of Toeplitz networks for various tasks in system identification and control are given in[1-7]. Here we discuss some of the general properties of Toeplitz networks that are relevant to all applications.

A typical configuration for a Toeplitz network is depicted in Figure 6. An externally provided training signal, ξ is first passed through a tapped delay line so that $\overline{\xi}$ is the basic input to the network. The output of some designated array constitutes the output to the network and this vector is compared with a reference signal $\overline{\eta}(n)$obtained by passing a reference signal $\eta(n)$ through a tapped delay line. $\eta(n)$ might be the output of some dynamic system with ξ as input. The difference, $\overline{\varepsilon}(n)$, is then injected into the backward signal path of the output array.

The formula for the update of any particular weight matrix can be obtained as a direct consequence of the definitions given in Figures 2a, 3a and 5a. First, we introduce supporting notation. Let * denote the Hadamard product of matrices (element-by-element multiplication). U_0 is a particular instance of a synaptic constraint matrix. In the following, let U_n denote the matrix:

$$U_n \in \mathbf{R}^{L\times L}$$
$$(U_n)_{kj} = \begin{cases} 0; j < k+n \\ 1; j \geq k+n \end{cases} \quad (1)$$

Furthermore, the symbols $\hat{g}(\cdot)$ and $\hat{g}'(\cdot)$ denote:

$$\hat{g}(\overline{x}) \in \mathbf{R}^L : (\hat{g}(\overline{x}))_\kappa \triangleq g(\overline{x}_\kappa)$$
$$\hat{g}'(u) \in \mathbf{R}^{L\times L} : \hat{g}(u) \triangleq \underset{\kappa=1\ldots L}{\text{diag}} \{g'(u_\kappa)\} \quad (2)$$

where g is the sigmoidal nonlinearity of the neurons (see Figure 2) and g' is its derivative. Suppose that $W_{\kappa j}$ is the weight matrix of the bundle of synapses that feeds the output of neural array j into the inputs of array κ. Let the outputs of array κ in the forward and backward paths, respectively, be x_κ and X_κ. Then as a consequence of our definitions:

$$x_\kappa = \hat{g}(U_\kappa) \quad (3)$$
$$U_\kappa = \sum_j W_{\kappa j} x_j + A_\kappa \overline{\xi} \quad (4)$$
$$X_\kappa = \hat{g}'(U_\kappa)\left(\sum_j W_{j\kappa}^T X_j + \Omega_\kappa \overline{\in} \right) \quad (5)$$

where;

$$A_\kappa = \begin{cases} 1, \text{ if array } \kappa \text{ is the input array} \\ 0, \text{ otherwise} \end{cases}$$
$$\Omega_\kappa = \begin{cases} 1, \text{ if array } \kappa \text{ is the output array} \\ 0, \text{ otherwise} \end{cases} \quad (6)$$

and where:

$$\overline{\in} = \overline{\eta} - \sum_\ell \Omega_\ell x_\ell \quad (7)$$

Equally obvious from our definitions (see Figure 3) and the above notation is that;

$$W_{\kappa j}(n+1) = W_{\kappa j}(n) + \mu(n) U_o * (X_\kappa x_j^T) \quad (8)$$

where $\mu(n)>0$ (for all n) is the time varying adaptive speed appearing in Figure 3. $\mu(n)$ is defined specifically below. However, before considering this matter we can readily show the following result:

Theorem 1

For the neural network composed as indicated in Fig. 6 and defined by relations (1)-(8), if we assume that all the vectors x_r and X_r are uniquely determined, then:

$$\frac{1}{\mu(n)}(W_{\kappa j}(n+1) - W_{\kappa j}(n)) = -\frac{\partial}{\partial W_{\kappa j}} \|\overline{\in}(n)\|^2 \quad (9)$$

Proof: Appendix A.

The above result indicates that the increment in each synaptic weight is proportional to the negative gradient of the square of the norm of the output error. Thus, any Teoplitz network set up as we have stipulated in Fig. 6a is a gradient decent machine.

With regard to $\mu(n)$, we first define

$$\mu(n) = \alpha F(n), \ \alpha > 0 \quad (10)$$

where we term α the *learning rate constant*. In gradient descent schemes, instability in the form of oscillatory divergence can result if the adaptive speeds are too large. In the present approach, it is desired to make $\mu(n)$ time varying so that such instability is avoided. In particular $F(n)$ provides a scaling of μ such that stability and convergence requirements are reduced to a restriction on α. Under broad conditions, α is a

"universal" constant in that it may be chosen once and for all (and built into the system) so as to secure desired convergence rates independently of the detailed characteristics of the training stimuli or of the systems to be identified and controlled. This confers the benefit of stable, autonomous performance.

The specific definition of $F(n)$ in (10) is:

$$F(n)=\frac{P(n)}{A(n)} \tag{11}$$

P = "performance function"

$$= L(\frac{1}{2}\|\overline{\in}(n)\|^2 - J^*) \tag{12}$$

where

$$L(x)\triangleq\begin{cases}x; & x\geq 0\\ 0; & x<0\end{cases} \tag{13}$$

$$J^* = \text{desired mean-square error level} \tag{14}$$

and

A = "neural activity level"

$$= \sum_{\text{all synapses}}(X_\kappa x_\kappa)^2 \tag{15}$$

where x_κ and X_κ denote the inputs on the forward path and on the backward path, respectively, to synapse κ. In particular, for the structure of Fig. 6:

$$A=\sum_{\kappa,j}\left\|U_o*\left(*X_\kappa x_j^T\right)\right\|_F^2 \tag{16}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. In the definition, (11), of $F(n)$, the numerator and denominator are derived from readily accessible signals. The numerator, P, signifies the performance requirement of the overall network in terms of the square of the norm of the output error, $\frac{1}{2}\|\overline{\in}(n)\|^2$. Note that by virtue of the function chosen, the system is not required to achieve a minimum value of $\frac{1}{2}\|\overline{\in}(n)\|^2$ but only to attain a value that falls below a desired level, J^*. Thus the network attempts to achieve "good-enough" performance, not necessarily optimal performance.

The above concludes our definition of the kind of modules that are used within the ANC architecture. In the following we consider some general characteristics of networks composed as in Fig. 6. The next section offers some simple results concerning convergence behavior of Toeplitz networks.

3. Some Convergence Properties of Toeplitz Networks

Here we consider some simple, introductory results on the convergence of Toeplitz networks of the type indicated in Fig. 6 and with the defining relations (1-8), (10-16). In particular, we trace the consequences for convergence of the time-varying adaptive speed adjustment, (11). The manner of convergence proof owes much to the work of Narendra in stable adaptive systems[31], except that in the present instance a discrete time system is being addressed.

To begin, not that $\frac{1}{2}\|\overline{\in}(n)\|^2$ is a function of all the weight matrices $W_{\kappa j},\kappa,j=1,\ldots,N_a$ (where N_a is the number of neural arrays) at time n and of the past history; $\{\xi(n-m),m=0,1,\ldots\}$ of the training stimulus. Alternatively, if we stack all of the nonzero and independent elements of the $W_{\kappa j}$'s in a single vector, $W(n)$; then $\frac{1}{2}\|\overline{\in}(n)\|^2$ is a function of $W(n)$ and $\{\xi(n-m),m=0,1,\ldots\}$. We symbolize this by writing:

$$\frac{1}{2}\|\overline{\in}(n)\|^2 = J(W(n),\ \{\xi(n-m);m=0,1,\ldots\}) \tag{17}$$

or, more briefly, $\frac{1}{2}\|\overline{\in}(n)\|^2 = J(W(n))$, when we wish to emphasize the functional dependence of $\|\overline{\in}\|^2$ on W. In (17) $W(n)\in\mathbf{R}^{N_s}$ where N_s is the is the total number of independent, nonzero, synaptic weights. Clearly, by (17), J is a nonnegative function of $W(n)$, i.e. $J(W(n))\geq 0$ for all $W\in\mathbf{R}^{N_s}$.

For bounded and convergent response, the crux of the matter is the "shape" of $J(n)$ as a function of $W\in\mathbf{R}^{N_s}$. In order to state the simplest result, recall that $F(x_1,x_2,\ldots,x_N)$ is termed a *homogeneous function of degree M* if for $\beta\in\mathbf{R}$:

$$F(\beta x_1,\beta x_2,\ldots,\beta x_N) = \beta^M F(x_1,x_2,\ldots,x_N) \tag{18}$$

By differentiating both sides with respect to β and sestting β equal to unity, we see that:

$$F=\frac{1}{M}\sum_{\kappa=1}^{N}x_\kappa\frac{\partial F}{\partial x_\kappa} \tag{19}$$

With the above terminology, we may define a slightly more general class of functions. We shall say that $J(W(n))$ is *bounded by a homogeneous function of degree M* iff:

$$0<\left((J-J_0)/\frac{1}{M}(W-W_0)^T\frac{\partial J}{\partial W}\right)\leq 1 \tag{20}$$

$$W_0\in\mathbf{R}^{Ns},\quad \forall(W-W_0)\neq 0_{Ns}$$

and J has a single global minimum at $W=W_0$.

Note that the network can succeed in accomplishing its task only when there is a minimum

of J that is smaller than the stipulated level, J^*, of acceptable error. Thus, we shall only consider the case in which $J(W_0) \le J^*$. The following preliminary result must be stated:

Lemma 1

If, together with the assumptions of Theorem 1, $J(W(n))$ is bounded by a homogeneous function of degree M and $J(W_0) \le J^*$ then, defining:

$$\tilde{W}(n) \triangleq W(n) - W_0 \tag{21}$$

the vector $\tilde{W}$ satisfies:

$$\tilde{W}(n+1) = \left[I_{Ns} - \frac{\alpha}{M} \delta(n) \Psi(n) \Psi^T(n) \right] \tilde{W}(n) \tag{22}$$

where:

$$\Psi(n) \triangleq \frac{\partial J(n)}{\partial W(n)} \Big/ \left\| \frac{\partial J(n)}{\partial W(n)} \right\| \tag{23}$$

$$\left(\therefore \|\Psi\| = 1 \right)$$

$$\delta(n) = \frac{L(J - J^*)}{J - J_0} \frac{J(n) - J_0}{\left(\frac{1}{M} \tilde{W}^T \frac{\partial J}{\partial W} \right)} \in [0,1] \forall n \tag{24}$$

Proof: Appendix B.

Equation (22) is similar to an equation arising frequently in linear adaptive control (see [31]). A complicating factor here is that the unit vector Ψ may also depend upon W. Nevertheless, it is easy to derive a sufficient condition for the boundedness of the time series $\left\{ W(n) \in \mathbf{R}^{Ns}; n = 0,1,\ldots \right\}$. We have:

Theorem 2

Under the assumptions of Lemma 1:

$$0 < \alpha < 2M \Rightarrow \left\| \tilde{W}(n+1) \right\| \le \left\| \tilde{W}(n) \right\| \forall n \tag{25}$$

Proof: We use (22) to evaluate $\left\| \tilde{W}(n+1) \right\|^2$ directly:

$$\begin{aligned} \left\| \tilde{W}(n+1) \right\|^2 &= \tilde{W}^T(n) \left[I - \frac{\alpha}{M} \delta(n) \Psi(n) \Psi^T(n) \right] \\ &\quad x \left[I - \frac{\alpha}{M} \delta(n) \Psi(n) \Psi^T(n) \right] \tilde{W}(n) \\ &= \tilde{W}^T(n) \left[I - 2 \frac{\alpha}{M} \delta(n) \Psi(n) \Psi^T(n) \right. \\ &\quad \left. + \left(\frac{\alpha}{M} \right)^2 \delta^2(n) \Psi(n) \Psi^T(n) \right] \tilde{W}(n) \\ &= \left\| \tilde{W}(n) \right\|^2 - \left(2 \frac{\alpha \delta(n)}{M} - \left(\frac{\alpha}{M} \right)^2 \delta^2(n) \right) \left(\Psi^T(n) \tilde{W}(n) \right)^2 \end{aligned} \tag{26}$$

where $\|\Psi(n)\| = 1$ has been used. Defining:

$$\cos\phi(n) \triangleq \Psi^T(n) \tilde{W}(n) / \left\| \tilde{W}(n) \right\| \tag{27}$$

where, obviously, $|\cos\phi(n)| \le 1 \forall n$, (26) becomes:

$$\begin{aligned} &\left\| \tilde{W}(n+1) \right\|^2 \\ &= \left[1 - \left(2 \frac{\alpha}{M} \delta(n) - \left(\frac{\alpha}{M} \delta(n) \right)^2 \right) \cos^2 \phi(n) \right] \left\| \tilde{W}(n) \right\|^2 \end{aligned} \tag{28}$$

Since $|\cos(n)| \le 1$, the term in brackets on the right is obviously nonnegative for all n. Finally, since both $|\cos(n)| \le 1$ and $\delta(n) \le 1$ for all n, the condition $\alpha < 2M$ implies that this term is less than or equal to unity. □

We note in connection with this result that, unless $\cos\phi(n) = 0$, the condition $\alpha < 2M / \delta(n)$ is necessary for $\left\| \tilde{W}(n+1) \right\| \le \left\| \tilde{W}(n) \right\|$. In any case, Theorem 2 shows that if the learning rate constant is selected less than twice the degree of homogeneity of J, then the sequence of $\tilde{W}(n)$ is bounded in norm. Moreover, as a practical matter, it often transpires that $\delta(n) = 1$ and $\cos\phi(n) \ne 0$ so that $\alpha < 2M$ is also necessary for nondivergence. Thus, it does not appear possible to improve the convergence speed of the network by increasing the learning rate constant beyond $2M$.

Theorem 2 shows a simple condition on the constant α that guarantees boundedness of the sequence of synaptic weights. However, the result does not prove convergence of the network output to some desirable level of error. We complete our investigation of the simple case assuming condition (20) with the following result.

Theorem 3

Under the assumptions of Lemma 1 and considering $\alpha < 2M$:

$$\lim_{n \to \infty} J(n) \le J^* \tag{29}$$

Proof: Rearranging (28):

$$\begin{aligned} &\left\| \tilde{W}(n+1) \right\|^2 - \left\| \tilde{W}(n) \right\|^2 \\ &= - \left(2 \frac{\alpha}{M} \delta(n) - \left(\frac{\alpha}{M} \delta(n) \right)^2 \right) \left(\Psi^T(n) \tilde{W}(n) \right)^2 \end{aligned} \tag{30}$$

This gives:

$$\left\|\tilde{W}(n)\right\|^2 - \left\|\tilde{W}(0)\right\|^2$$
$$= \sum_{k=0}^{n-1}\left(\left\|\tilde{W}(k+1)\right\|^2 - \left\|\tilde{W}(k)\right\|^2\right) \qquad (31)$$
$$= -\sum_{k=0}^{n-1}\left[2\frac{\alpha}{M}\delta(k) - \left(\frac{\alpha}{M}\delta(k)\right)^2\right]\left(\Psi^T(k)\tilde{W}(k)\right)^2$$

However, by virtue of $\alpha<2M$ and Theorem 2, $\left\|\tilde{W}(n)\right\|$ is bounded for all n. Hence:

$$\lim_{n\to\infty}\sum_{k=0}^{n}\left[2\frac{\alpha}{M}\delta(k) - \left(\frac{\alpha}{M}\delta(k)\right)^2\right]\left(\Psi^T(k)\tilde{W}(k)\right)^2 < \infty \qquad (32)$$

Thus, the summand is a bounded, nonnegative ℓ_2 sequence. The only limit point is zero:

$$\lim_{k\to\infty}\left(2\frac{\alpha}{M}\delta(k) - \left(\frac{\alpha}{M}\delta(k)\right)^2\right)\left(\Psi^T(k)\tilde{W}(k)\right)^2 = 0 \qquad (33)$$

There are now several possibilities: either the first factor converges to zero or the second factor does so or both converge to zero. Consider the first case. This is possible only if $\lim_{k\to\infty}\frac{\alpha}{M}\delta(k)$ equals either zero or two. Since $\frac{\alpha}{M}<2$ and $\delta(k)\le 1$ the limiting value of two is ruled out. Hence the convergence of the first factor in (33) to zero implies $\lim_{k\to\infty}\delta(k)=0$, or, using (24):

$$\lim_{k\to\infty}\frac{L(J-J^*)}{J-J_0}\left(\frac{J(k)-J_0}{\frac{1}{M}\tilde{W}^T\frac{\partial J}{\partial W}}\right) = 0 \qquad (34)$$

Because of the left-hand inequality in (20), this implies.

$$\lim_{k\to\infty}\frac{L(J(k)-J^*)}{J(k)-J_0} = 0 \qquad (35)$$

However, the fact that J has a single global minimum, J_0, and $J_0<J^*$ means that $L(J-J^*)/(J-J_0)$ can vanish only if $J\le J^*$. Therefore, the convergence of the first factor in (33) to zero implies (29).

Next consider the possibility that the second factor in (33) converges to zero. In view of the definitions (21) and (23) this implies $\lim_{k\to\infty}(W-W_0)^T\frac{\partial J}{\partial W}=0$. Because of the right hand inequality in (20), it follows that $\lim_{k\to\infty}(J-J_0)\le\frac{1}{M}\lim_{k\to\infty}(W-W_0)^T\frac{\partial J}{\partial W}=0$. Thus, since $J-J_0$ is nonnegative by assumption, $\lim_{k\to\infty}J(k)=J_0\le J^*$. Obviously if both factors in (33) converge to zero then the same conclusion follows. Therefore (33) implies (29). □

The above results, in so far as they are based upon (20), are very restricted but they do illustrate the importance of the adaptive speed formula, (11)-(15), for boundedness and convergence behavior. Also the last theorem tends to justify the notion that α can be treated as a "universal" constant — i.e. that a suitable value of α can be chosen once and for all and built in to the network and convergence can then be guaranteed for a broad class of tasks. Further, the condition $\alpha < 2M$ often provides a reasonably firm, practical guideline in the design of Toeplitz networks. In many instances, J is either locally or even globally approximated by a homogeneous formation of degree M. Thus, in practice, for a given class of problems, one estimates a lower bound on the degree of homogeneity and chooses $\alpha < 2M$. Numerical and experimental experience (that will not be reviewed here) seems to indicate that $\alpha = M$ is, on the whole, the "best" choice — i.e. convergence is fastest and oscillatory behavior does not occur.

The assumptions underlying above results are very restricted and much remains to be done. In particular it is of interest to determine the behavior of the network when J, considered as a function of the weights, has several local minima. On this issue, no rigorous results are forthcoming, but in the next section we do offer an intriguing conjecture based on numerical experience.

4. A Conjecture Regarding Instability Local Minima

In general nonlinear problems, the function $J(W(n))$ is not globally convex — i.e. has several local minima rather than a single global minimum. This situation is illustrated schematically in Figure 7 where W is as defined above. Neural networks heretofore employed to solve optimization or system identification problems generally experience the difficulty that their solution can be trapped in one of the high lying local minima — i.e. points A,B,C,D or E in Figure 7 — thus failing to achieve satisfactory performance (that is: failing to converge to minima that lie below the acceptable performance tolerance, J^* in Figure 7). The system discussed here however, appears to exhibit quite different behavior. Based on particular examples, we conjecture that because of the structure of the adaptive speed function in (11-15) high-lying minima (points A-E) are unstable in general. In other words, if the weights, W, are initially in the vicinity of point C for example, the system executes ever larger departures from C. We further conjecture that within finite time the network leaves the "valley" having point C as its bottom and

ultimately enters one of the "valleys" associated with minima below J^*. Once this occurs (and assuming $\alpha < 2M$ for each of these low-lying valleys), the system converges to a value of W within the low-lying valley and the final performance, J is less than J^*. If generally true, this feature would make the present system very attractive for applications involving complex nonlinear optimization problems that have numerous local minima.

Some evidence that local minima larger than J^* are unstable can be gleaned from simple examples. Consider $W \in \mathbf{R}^1$ and

$$J = J_0 + W^2 \tag{36}$$

Obviously J-J_0 is homogeneous of degree 2 and if $J_0 < J^*$, the origin is stable for $\alpha < 4$. However, for $J_0 > J^*$, the counterpart to (28) is:

$$|W(n+1)| = \left|1 - \frac{\alpha}{2} - \frac{\alpha(J_0 - J^*)}{2|W|^2}\right| |W(n)| \tag{37}$$

In this case, even for previously stable values of α the origin is unstable. Actually there is a repelling singularity at the origin. Numerical simulations of (37) show $|W|$ going through a succession of episodes wherein $|W|$ first approaches the zero smoothly then is abruptly "bounced" away from zero. Trajectories are very sensitive to initial conditions.

A slightly more complex example serves to show that not only are high lying local minima unstable, but the system does eventually settle into one of the minima for which $J<J^*$. Consider again, $W \in \mathbf{R}^1$, but with:

$$J = \frac{1}{2}W^2 + (\sin \pi W)^2 \tag{38}$$

As illustrated in Figure 8, this error function has seven local minima, including the global minimum at W=0. The counterpart to (22) is:

$$W(n+1) = W(n) - \alpha \frac{L(J - J^*)}{W + \pi \sin(2\pi W)} \tag{39}$$

Taking J^*=0 and α =1.0 we see that, according to our conjecture, only the origin can be stable. With these values, Figure 9 shows a superposition of $W(n)$ for 120 different initial values of W distributed evenly over [0, 15] in increments of 0.125 (obviously trajectories for negative initial values are the reflections about the ordinate). It is seen that although the system spends a larger fraction of time around the local minima near W=1 and W=2, all trajectories eventually terminate at W=0, which is the one minimum where $J \le J^*$. If one tries similar experiments with $J^* > 0$, then the system is found to settle into one of the (possibly several) local minima for which $J \le J^*$.

These and similar examples provide incentive for the rigorous formulation and proof of the above conjectures in future work.

5. Conclusion

In this paper we have given a self-contained description of the Adaptive Neural Control (ANC) parallel processing architecture for identification and control. Fundamental results on convergence were given for a simple situation which tends to show that network stability can be built in by appropriate prior choice of the learning rate constant. This result is essentially due to the time-varying adaptive speed update which is a novel feature of the architecture. Moreover, numerical experience suggests that ANC systems do not settle into high-lying local minima of the error function. This intriguing conjecture is the object of further invesigation.

References

1. D.C. Hyland, "Neural network architecture for on-line system identification and adaptively optimized control," *Proceedings of the IEEE conference on Decision and Control.*, Brighton, U.K., December 1991.
2. D.C. Hyland and J.A. King, "Neural network architectures for stable adaptive control, rapid fault detection and control system recovery," *Proceedings of the 15th Annual AAS Guidance and Control Conference*, Keystone, CO, February 1992.
3. D.C. Hyland, "Adaptive neural control architecture: a tutorial," *Proceedings of the Industry, Government and University Forum on Adaptive Neural Control for Aerospace Structural Systems*, Melbourne, FL, August 1993.
4. D.C. Hyland, "Adaptive neural control for flexible aerospace systems: Progress and prospects," *Proceedings of the 10th IEEE International Symposium on Intelligent Control 1995*, Monterey, CA, 27-29 August 1995.
5. D.C. Hyland and J.A. King, "Decentralized adaptive neural control for distributed mesoscale actuators and sensors," *Proceedings of the 1995 SPIE Conference on Smart Structures and Materials, Technical Conference 2442*, San Diego, CA, 26 February - 3 March, 1995.
6. D.C. Hyland and L.D. Davis, "A Multiple-input, multiple-output neural architecture for the suppression of a multi-tone disturbance," *19th Annual AAS Guidance and Control Conference*,

Paper No. AAS 96-067, Breckenridge, Colorado, February 7-11, 1996.

7. D.C. Hyland, L.D. Davis, A. Das and G. Yen, "Antonomous neural control for structure vibration suppression," AIAA Guidance, Navigation and Control Conference, Paper No. AIAA-96-3923, San Diego, CA, July 29-31, 1996.
8. W.S. McCulloch, and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics* 5, 115-133, 1943.
9. D.O. Hebb, "The organization of behavior: A neurophsychological theory," New York: Wiley, 1949.
10. F. Rosenblatt, "The perception: A probabilistic model for information storage and organization in the brain," *Psychological Review* 65, 386-408, 1958.
11. F. Rosenblatt, "Perceptron simulation experiments," *Proceedings of the Institute of Radio Engineers*, 48, 301-309, 1960.
12. F. Rosenblatt, "On the convergence of reinforcement procedures in simple perceptrons," Report VG-1196-G-4. Cornell Aeronautical Laboratory, Buffalo, NY, 1960.
13. F. Rosenblatt, *Principles of Neurodynamics*, Washington, DC: Spartan Books, 1962.
14. B. Widrow and M.E. Hoff, Jr., "Adaptive switching circuits," *IRE WESCON Convention Record*, 96-104, 1960.
15. B. Widrow and M.A. Lehr, 1990, "30 years of adaptive neural networks: Perceptron, madaline, and backprogagation," *Proceedings of the IEEE* 78, 1415-1442, 1990.
16. B. Widrow and S.D. Stearns, *Adaptive Signal Processing*, Englewood Cliffs, NJ: Prentice-Hall, 1985.
17. P.J. Werbos "Beyond regression: New tools for prediction and analysis in the behavioral sciences." Ph.D. Thesis, Harvard University, Cambridge, MA, 1974.
18. D.B. Parker "Learning-logic: Casting the cortex of the human brain in silicon," Technical Report TR-47. Center for Computational Research in Economics and Management Science, MIT, Cambridge, MA, 1985.
19. Y. LeCun. "Une procedure d'apprentissage pour reseau a seuil assymetrique," Cognitiva 85, 599-604, 1985.
20. D.E. Rumelhart, G.E. Hinton and R.J. Williams, "Learning representations by back-propagating errors," *Nature (London)*, 323, 533-536, 1986.
21. D.E. Rumelhart, G.E. Hinton and R.J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* (D.E. Rumelhart and J.L. McClelland, eds.) Vol. 1, chapter 8, Cambridge, MA: MIT Press, 1986.
22. D.E. Rumelhart and L.J. McClelland, eds., *Parallel distributed processing: Explorations in the Microstructure of Cognition*, Vol. 1 Cambridge, MA: MIT Press, 1986.
23. K.J. Lang and G.E. Hinton, "The development of the time-delay neural network architecture for speech recognition," Technical Report CMU-CS-88-152. Carnegie-Mellon University, Pittsburgh, PA, 1988.
24. A. Waibel, T. Hanazawa, G. Hinton, K. Shikano and K.J. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Transactions on Acoustics, Speech, and Signal Processing* ASSP-37, 328-339, 1989.
25. E.A. Wan, "Time series prediction by using a connectionist network with internal delay lines," *In Time Series Prediction: forecasting the Future and Understanding the Past* (A.S. Weigend and N.A. Gershenfeld, eds), 195-217. Reading, MA: Addison-Wesley, 1994.
26. P.J. Werbos "Backpropagation through time: What it does and how to do it," *Proceedings of the IEEE*, 78, 1550-1560, 1990.
27. E.A. Wan, "Temporal backpropagation for FIR neural networks," *IEEE International Joint Conference on Neural Networks*, Vol. 1, 575-580, San Diego, CA, 1990.
28. E.A. Wan "Temporal backpropagation: An efficient algorithm for finite impulse response neural networks," *In Proceedings of the 1990 Connectionist Models Summer School* (D.S. Touretzky, J.L. Elman, T.J. Sejnowski, and G.E. Hinton, eds.), 131-140, San Mateo, CA: Morgan Kaufmann, 1990.
29. R.J. Williams and D. Zipser "A learning algorithm for continually running fully recurrent neural networks," *Neural Computation* 1, 270-280, 1989.
30. K.S. Narendra and K. Parthasarathy "Identification and control of dynamical systems using neural networks," *IEEE Transactions on Neural Networks* 1, 4-27, 1990.
31. K.S. Narendra and A.M Annaswamy, *Stable Adaptive Systems*, Englewood Cliffs, NJ: Prentice-Hall, 1989.

Appendix A

Proof of Theorem 1

First, we evaluate the derivative of $\frac{1}{2}\|\bar{\in}\|^2$ with respect to $(W_{kj})_{\ell m}$ — where this denotes the $(\ell,m)^{th}$ element of W_{kj}. If this element is constrained to be zero because W_{kj} is constrained to be upper triangular, then the increment is zero. Therefore, consider the nontrivial case where $\ell \geq m$. Using (7) we first obtain:

$$-\frac{\partial}{\partial(W_{kj})_{\ell m}}\frac{1}{2}\|\bar{\in}\|^2 = \bar{\in}^T \sum_r \Omega_r \frac{\partial x_r}{\partial(W_{kj})_{\ell m}} \qquad (A.1)$$

Next, we use (3) to evaluate $\frac{\partial x_r}{\partial(Wkj)_{\ell m}}$:

$$\frac{\partial x_r}{\partial(W_{kj})_{\ell m}} = \hat{g}'(U_r)\left(\sum_s W_{rs}\frac{\partial x_s}{\partial(W_{kj})_{\ell m}} + \delta_{rk}e^{(\ell)}e^{(m)T}x_j\right) \qquad (A.2)$$

where $e^{(\ell)}$ is the unit coordinate vector:

$$\left(e^{(\ell)}\right)_m = \delta_{\ell m} \qquad (A.3)$$

Also, define:

$$\hat{G}' = \text{block}-\text{diag}\left\{\hat{g}'(U_k)\right\} \qquad (A.4)$$

$$W = \begin{bmatrix} W_{11} & W_{12} & \cdots & W_{1N} \\ W_{21} & W_{22} & \cdots & W_{2N} \\ \vdots & & & \vdots \\ W_{N1} & W_{N2} & \cdots & W_{NN} \end{bmatrix} \qquad (A.5)$$

and let $E^{(\ell)}$ denote:

$$E^{(\ell)} = \begin{bmatrix} O_{n_1\times n_\ell} \\ O_{n_2\times n_\ell} \\ \vdots \\ I_{n_\ell\times n_\ell} \\ \vdots \\ O \end{bmatrix} \leftarrow \ell^{th}\ \text{subblock} \qquad (A.6)$$

where n_ℓ denotes the dimension of x_ℓ with $N = \sum_\ell n_\ell$.

With the above notation, we note that the assumption that all the x_r's and the X_r's are determinate means that $(I_N - \hat{G}'W^T)^{-1}$ exists, because otherwise the X_r's would be indeterminate. This also implies the nonsingularity of $(I_N - \hat{G}'W)$. Hence, from (A.2), we obtain:

$$\frac{\partial x_r}{\partial(W_{kj})_{\ell m}} = E^{(r)T}(I-\hat{G}'W)^{-1}\hat{G}'E^{(k)}e^{(\ell)}(x_j)_m \qquad (A.7)$$

Therefore, returning to (A.1), we have:

$$-\frac{\partial}{\partial(W_{kj})_{\ell m}}\frac{1}{2}\|\bar{\in}\|^2$$

$$= \bar{\in}^T \sum_r \Omega_r E^{(r)T}(I-\hat{G}'W)^{-1}\hat{G}'E^{(k)}e^{(\ell)}(x_j)_m$$

$$= \left(\sum_r E^{(k)T}\hat{G}'(I-W^T\hat{G}')^{-1}E^{(r)}\Omega_r\,\bar{\in}\right)_\ell (x_j)_m$$

Obviously since $\hat{G}'(I-W^T\hat{G}')^{-1} = (I-\hat{G}W^T)^{-1}\hat{G}'$ we obtain:

$$-\frac{\partial}{\partial(W_{kj})_{\ell m}}\frac{1}{2}\|\bar{\in}\|^2 = (E^{(k)T}(I-\hat{G}'W^T)^{-1}\hat{G}'\sum_r E^{(r)}\Omega_r\,\bar{\in})_\ell (x_j)_m \qquad (A.8)$$

However, referring to (5), it is apparent that:

$$X_k = E^{(k)T}(I-\hat{G}'W^T)^{-1}\hat{G}'\sum_r E^{(r)}\Omega_r\,\bar{\in} \qquad (A.9)$$

Therefore, on comparing this with (A.8), we conclude that:

$$-\frac{\partial}{\partial(W_{kj})_{\ell m}}\frac{1}{2}\|\bar{\in}\|^2 = (X_k)_\ell (x_j)_m \qquad (A.10)$$

for $\ell \geq m$. Considering all ℓ and m and interpreting $\frac{\partial}{\partial W_{kj}}\frac{1}{2}\|\bar{\in}\|^2$ to mean that its $(\ell,m)^{th}$ element is $\frac{\partial}{\partial(W_{kj})_{\ell m}}\frac{1}{2}\|\bar{\in}\|^2$, we obtain:

$$-\frac{\partial}{\partial W_{kj}}\frac{1}{2}\|\bar{\in}\|^2 = U_0 * (X_k x_j^T) \qquad (A.11)$$

However, by (8), the right hand side of (A.11) is simply $\frac{1}{\mu}(W_{kj}(n+1)-W_{kj}(n))$. This completes the proof. □

Appendix B

Proof of Lemma 1

Noting that W is the vector wherein all the nonzero elements of the W_{kj}'s are stacked, (9) implies that:

$$\frac{1}{\mu(n)}(W(n+1)-W(n)) = -\frac{\partial}{\partial W}\frac{1}{2}\|\bar{\in}(n)\|^2 = -\frac{\partial}{\partial W}J(W(n)) \qquad (B.1)$$

Expressing this in terms of $\tilde{W}$:

$$\tilde{W}(n+1) = \tilde{W}(n) - \mu(n)\frac{\partial J}{\partial W} \qquad (B.2)$$

Using (10), (11) and (12):

$$\tilde{W}(n+1) = \tilde{W}(n) - \alpha\frac{L(J-J^*)}{A(n)}\frac{\partial J}{\partial W} \qquad (B.3)$$

Now, comparison of (8) and (9) shows that $\frac{\partial J}{\partial W_r} = -X_r x_r$ where x_r and X_r denote the inputs on the forward and backward signal paths, respectively, to synapse r. Then (15) implies:

$$A = \sum_r \left(\frac{\partial J}{\partial W_r} \right)^2 = \left\| \frac{\partial J}{\partial W} \right\|^2 \qquad \text{(B.4)}$$

Note also that since (20) is assumed, we can write:

$$J(n) - J_0 = \theta(n) \frac{1}{M} \tilde{W}^T \frac{\partial J}{\partial W} \qquad \text{(B.5)}$$

where

$$\theta(n) \in (0,1] \forall n \qquad \text{(B.6)}$$

We can substitute (B.4) and (B.5) into (B.3) to get:

$$\tilde{W}(n+1) = \tilde{W}(n) - \alpha \frac{1}{\left\| \frac{\partial J}{\partial W} \right\|^2} \frac{L(J - J^*)}{J - J_0} \theta(n) \frac{1}{M} \tilde{W} \frac{T \partial J}{\partial W} \frac{\partial J}{\partial W} \qquad \text{(B.7)}$$

$$= \tilde{W}(n) - \frac{\alpha}{M} \frac{L(J - J^*)}{J - J_0} \theta(n) \Psi(n) \Psi^T(n) \tilde{W}$$

where the last line follows by noting $\tilde{W}^T \frac{\partial J}{\partial W} = \left(\frac{\partial J}{\partial W} \right)^T \tilde{W}$ and by using definition (23). Making use of (24) in (B.7) yields (22). Notice that since $J_0 = J(W_0) \le J^*$, $L(J - J^*) \le J - J_0$ for all $W(n) \in \mathbf{R}^{Ns}$. Hence in view of (20) we may conclude that $\delta(n) \in [0,1]$ for all n. This completes the proof. □

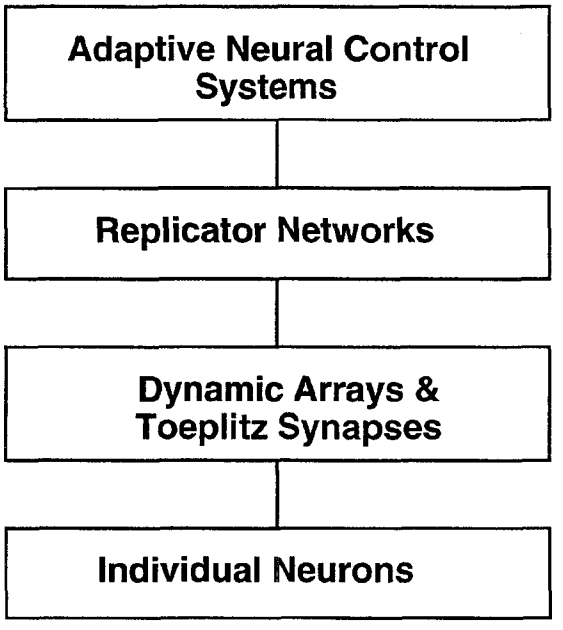

Figure 1. Hierarchy of Modular Neural Structures Progressing from Basic Constituents to Higher-Level Modules.

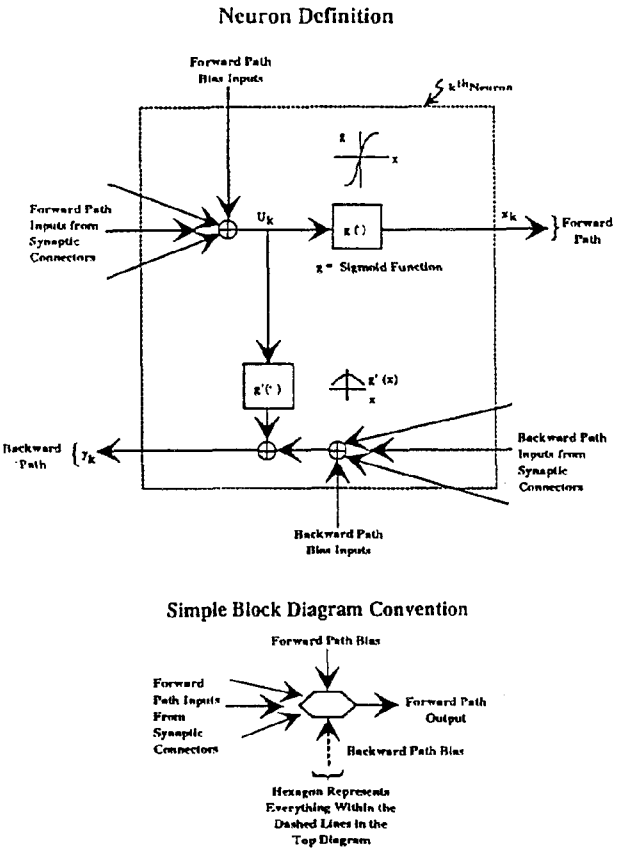

Figure 2. Definition of Basic Neuron, (a); Simplified Diagram Convention, (b).

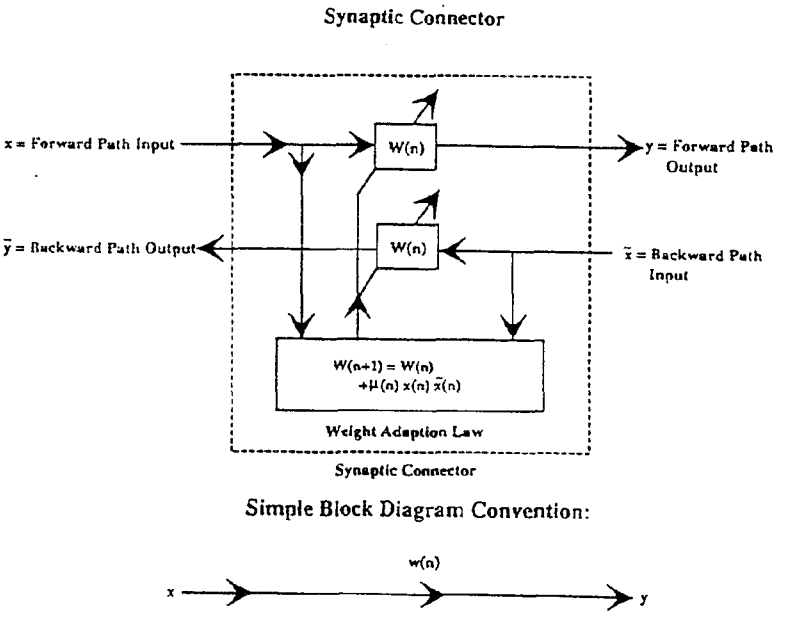

Figure 3: Synaptic Connector (a); Simplified Diagram, (b).

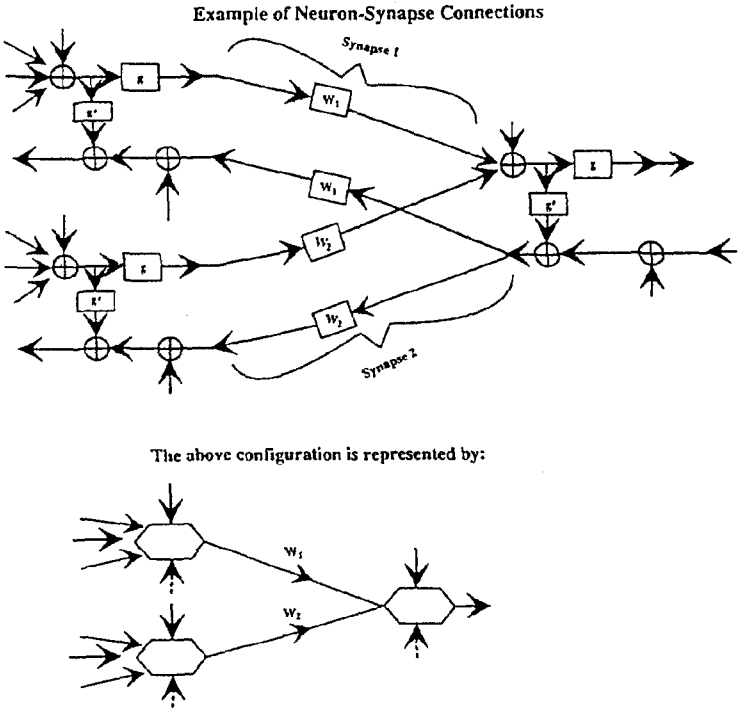

Figure 4: Example of Neuron-Synapse Connections: Detailed Diagram, (a); Simplified Block Diagram (b).

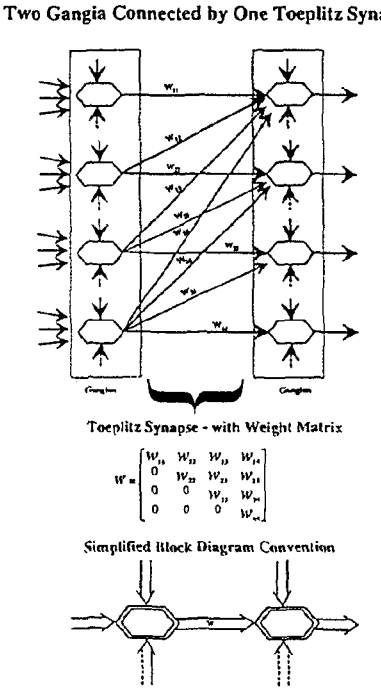

Figure 5: Two Dynamic Arrays Connected by One Toeplitz Synapse. Detailed Diagram (a); Simplified Block Diagram Convention, (b).

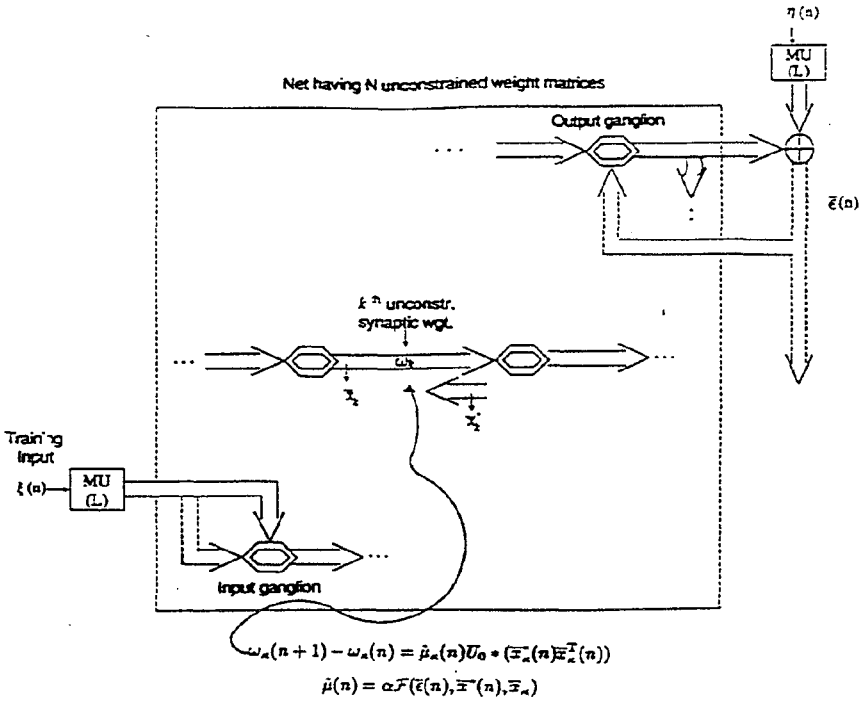

Figure 6: Summary of the Standard Setup for the Toeplitz Network Involving Arbitrary Interconnections of Dynamic Arrays.

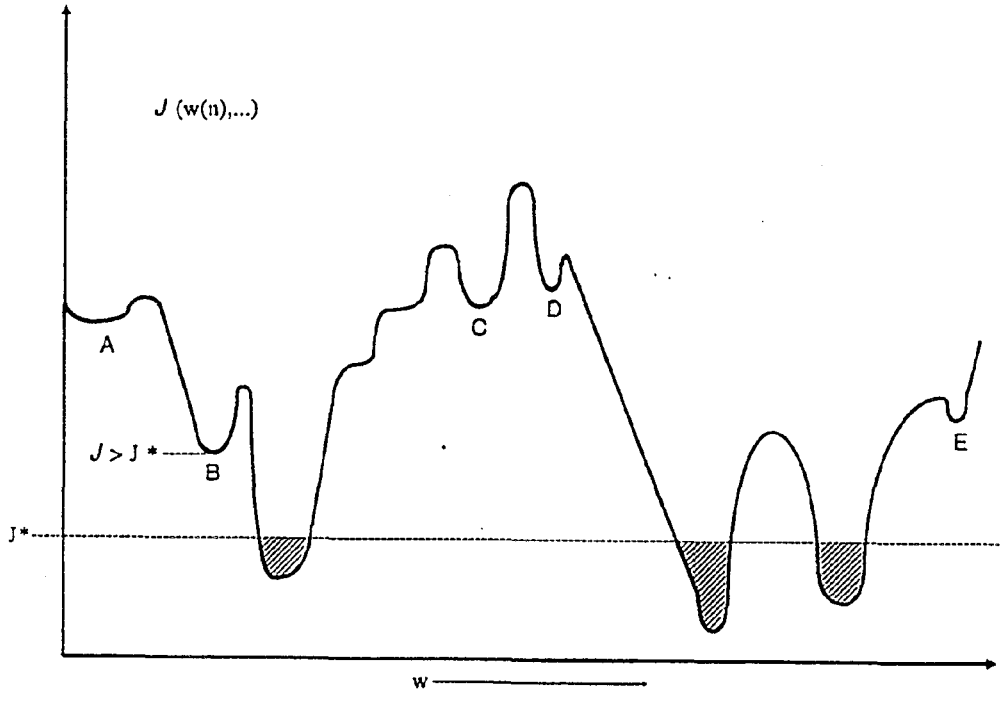

Figure 7: Global Convergence Conjecture: The System Never Comes to Rests in High-Lying Local Minima Such as Regions A-E. Instead, Such Regions are Exited in Finite Time and the System Winds Up in One of the Shaded Regions for Which $J \leq J^*$.

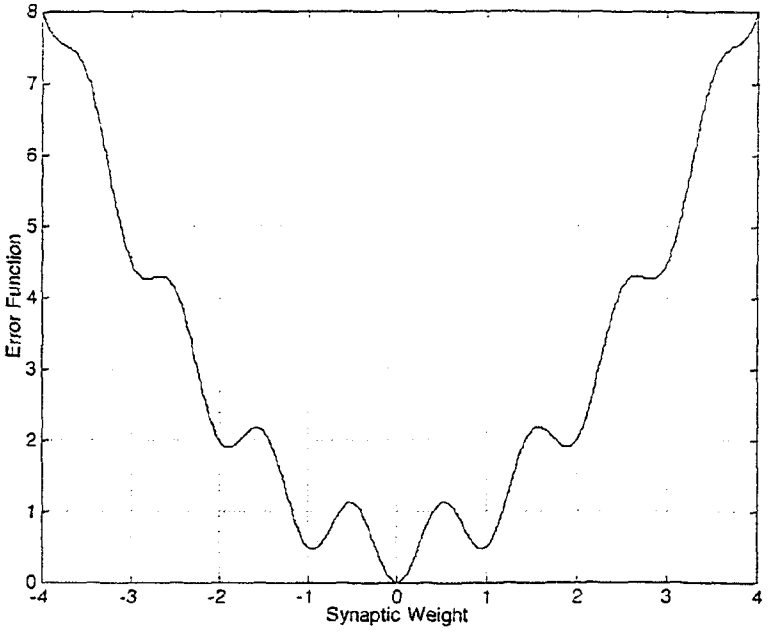

Figure 8: Penalty Function for the Example of Equation (38) Displays Seven Local Minima.

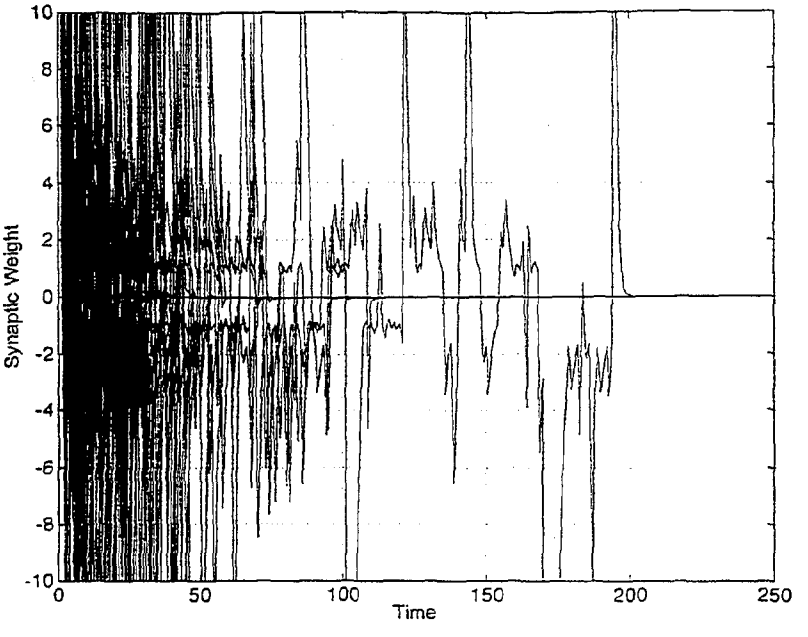

Figure 9: Superposition of 120 Trajectories of the System of Equation (38) Corresponding to Initial Values Along the Interval [0, 15], Spaced 0.125 Apart. $\left(\alpha = 1.0, J^* = 0\right)$.