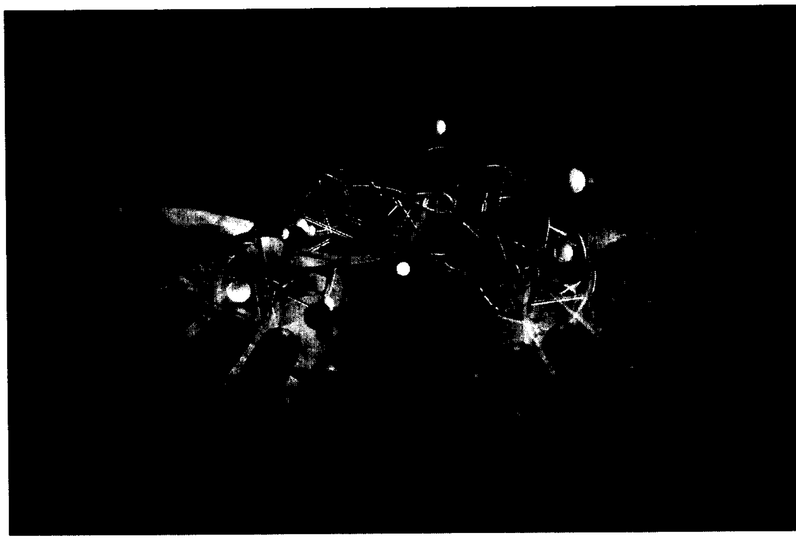
Qualitative Limitations Incurred in Implementations of Recurrent Neural Networks

Anthony N. Michel, Kaining Wang, Derong Liu, and Hui Ye

During the implementation process of artificial neural networks, deviations from the desired ideal neural network are frequently introduced. These include *parameter perturbations, transmission delays*, and *interconnection constraints*. In the present article, we study the effects of these realities of imperfection on the qualitative behavior of artificial feedback neural networks. To accomplish this, we utilize a specific class of neural networks (Hopfield-like neural networks) with a specific application (the realization of associative memories) as a vehicle for our study. The principal issues which we address concern the effects of parameter perturbations, transmission delays, and interconnection constraints on the accuracy and on the qualitative properties of the network memories.



Introduction

An increasing number of applications of artificial neural networks in a variety of areas has brought to the forefront problems that arise in the *implementation* of such networks, be it by VLSI, specialized digital hardware, opto-electronic means, and even simulations. Specifically, in such implementations, several limitations are encountered which may affect the qualitative performance of the neural networks, including limitations due to neural network *parameter uncertainties, interconnection constraints*, and *time delays* (especially in VLSI). One or more of these limitations are encountered in most applications that employ artificial neural networks, especially feedback neural networks, which are also called recurrent neural networks.

In this article we present a systematic analysis of artificial feedback neural networks which addresses limitations to the qualitative behavior (performance) of such networks which are due to parameter uncertainties, time delays, and constraints in the interconnecting structure of a network. Our objective is to present results which are as universal as possible. However, to fix ideas and provide motivation, we will employ specific classes of neural networks and specific applications. These include continuous-time (i.e., analog) Hopfield neural networks with applications to associative memories, using sigmoidal functions or saturation non-linearities as activation functions. The problems which we will address are as follows:

(i) Given a neural network with a desired set of operating points (e.g., in associative memories, a desired set of asymptotically stable equilibria which are used to store a desired set of stable memories), and given an associated neural network with *perturbations in the parameters*, determine conditions which will ensure that the perturbed neural network will possess a set of operating points which are near the operating points of the original unperturbed neural network. Determine sharp estimates of the distance between the operating points of the unperturbed neural network and the corresponding operating points of the perturbed neural network.

(ii) Given a neural network with a desired set of operating points (and with no time delays), and given an associated neural network with time delays, determine conditions which ensure that for all time delays that are less than some upper bound, the

This work was supported in part by the National Science Foundation under Grant ECS91-07728. *A.N. Michel and H. Ye are with the Department of Electrical Engineering, University of Notre Dame, Note Dame, IN 46556. K. Wang is with the Department of Electrical and Computer Engineering, Wayne State University, Detroit, MI 48202. D. Liu is with the Electrical and Electronics Research Department, General Motors, NAO R&D Center, Warren, MI 48090.*

0272-1708/95/$04.00©1995IEEE

operating points of the neural networks with time delays will coincide with the operating points of the corresponding network without delays and exhibit similar qualitative behavior (e.g., in associative memories, the network with delays should possess asymptotically stable equilibria which coincide with the asymptotically stable equilibria of the corresponding network without delays). Determine a sharp estimate for the above upper bound for the time delays.

(iii) Devise synthesis techniques (for associative memories) which incorporate constraints on the interconnecting structure of a network (e.g., prespecified sparsity constraints).

The issues addressed above have significant practical implications in VLSI implementations. Specifically, parameter perturbations will in general result in inaccuracies of the network operating points (stable memories), transmission delays will in general impose limitations on the size of a network, and interconnection constraints will in general result in a reduction of the storage capacity of a network. For a good reference which touches these issues, refer to Mead [19].

As mentioned earlier, we utilize implementations of associative memories by Hopfield-type networks as a vehicle for our presentation. This choice was deliberate, since this class of networks is well understood and is frequently used as a benchmark. It is not our intention here to compare such networks with other types of nets which are usually assumed to be more effective implementations of associative memories (e.g., implementations of Hamming distance networks).

Neural Network Models

We consider a class of artificial feedback neural networks which can be described by systems of first order ordinary differential equations given by

$$\dot{x} = -Bx + TS(x) + U \tag{1}$$

where x is a real n-vector (denoting the neuron variables), $\dot{x}$ denotes the time derivative of x, B is a real $n \times n$ diagonal matrix with positive elements (representing self-feedback), T is a real $n \times n$ matrix (representing the interconnections among the neurons), U is a real n-vector (representing bias terms), and the real n-vector valued function $S(x)$ (representing the neurons) will assume one of the following two forms:

(i) each component $s_i(x_i)$ of $S(x) = [s_1(x_1), ..., s_n(x_n)]^T$ is a *sigmoidal function* (i.e., s_i maps the real numbers R into the real interval (-1,1), it is smooth and monotonically increasing, and its graph is symmetric with respect to the origin); or

(ii) each component of $S(x)$ is a saturation function defined by

$$sat(x_i) = \begin{cases} 1, & x_i > 1 \\ x_i, & -1 \le x_i \le 1 \\ -1, & x_i < -1. \end{cases}$$

For the case when the *activation functions* $s_i(x_i)$ are sigmoidal functions and the matrix T is symmetric, system (1), which is frequently referred to as the *Hopfield model*, has been widely studied (see, e.g., [7], [10-13], [20], [21], [25-27], [32]). (In the present article, we will state explicitly when assuming that T is symmetric.) Among other applications, when the components of $S(x)$ are saturation functions, system (1) has been used to store bipolar memories and as cellular neural networks (see, e.g., [3], [4], [14-16]).

In the implementation of artificial neural networks (e.g., in VLSI implementations), parameter errors are often unavoidably encountered. We will use a perturbation model of system (1) given by

$$\dot{x} = -(B + \Delta B)x + (T + \Delta T)[S(x) + \Delta S(x)] + (U + \Delta U) \tag{2}$$

where $B, T, S(x)$ and U are defined as in (1), $\tilde{B} = B + \Delta B$ is a real $n \times n$ diagonal matrix with positive elements, $T + \Delta T$ is a real $n \times n$ matrix, $U + \Delta U$ is a real n-vector, and ΔB, ΔT, and ΔU are perturbation terms. When the s_i are saturation functions, we assume that $\Delta S(x) = 0$. When the s_i are sigmoidal functions, we assume that all components of $S(x) + \Delta S(x) = (S + \Delta S)(x)$ are also sigmoidal functions and we regard $\Delta S(x)$ as a perturbation term. We view (2) as a *perturbation model* for system (1).

In implementations of artificial neural networks, time delays are also frequently unavoidably encountered. Using once more system (1) as a vehicle for study, we will consider neural networks with transmission delays, described by a system of delay equations given by

$$\dot{x}_i(t) = -b_i x_i(t) + \sum_{j=1}^{n} t_{ij}^{(0)} s_j(x_j(t)) + \sum_{k=1}^{K} \sum_{j=1}^{n} t_{ij}^{(k)} s_j(x_j(t - \tau_k)) + U_i, \; i = 1, ..., n \tag{3}$$

where $x = (x_1, ..., x_n)^T$, $B = diag[b_1, ..., b_n]^T$, $U = (U_1, ..., U_n)^T$, where x, B, and U are defined similarly as in (1), where the $t_{ij}^{(k)}$ denote the neuron interconnections which are associated with delay $\tau_k > 0$, $k = 1, ..., K$, and where the $t_{ij}^{(0)}$ denote the neuron interconnections for which there are no time delays, $1 \le i, j \le n$. We will view (3) and (1) as having identical parameters, except for time delays.

A special case of system (3), having identical time delays $\tau_k = \tau$, $k = 1, .., K$, is given by

$$\dot{x}(t) = -Bx(t) + T_1 S(x(t)) + T_2 S(x(t - \tau)) + U \tag{4}$$

where the real $n \times n$ matrix T_1 represents the neuron interconnections with no delays while T_2 represents the neuron interconnections which are encumbered with a delay $\tau > 0$.

Some Background Material

A great deal of the qualitative behavior of artificial feedback neural networks concerns the stability properties of equilibria which serve in a variety of applications as the operating points of the networks. For example, in the case of associative memories, asymptotically stable equilibria of neural networks are used to generate stable memories.

When using the term *stability*, we will have in mind the concept of Lyapunov stability of an equilibrium. For purposes of completeness, we provide here heuristic explanations for some of the concepts associated with the Lyapunov theory. The precise delta-epsilon (δ-ε) definitions of these notions can be found, for example, in Chapter 5 of [23].

The neural network models discussed in the previous section are described either by a system of first-order ordinary differential equations (equations (1) or (2)) or by a system of first-order delay differential equations (equations (3) or (4)). These equations describe the process by which a system changes its state. To simplify our presentation, the focus in the following discussion will be on system (1) and (4). We first note that system (1) is defined on R^n, the real n-dimensional vector space with some norm $|\cdot|$ defined on it (e.g., the Euclidean norm) while system (4) is defined on the function space C_τ, consisting of all real, n-vector valued continuous functions defined on a real interval of length τ (e.g., $[t_0 - \tau, t_0]$, $\tau > 0$) with norm given by $\|x\| = \max_{t_0 - \tau < t < t_0} |x(t)|$ where $|\cdot|$ denotes some norm on R^n. For each real n-vector x_0, we let $\phi(t, t_0, x_0)$, $t \geq t_0$, denote the unique solution of system (1) with initial condition $\phi(t_0, t_0, x_0) = x_0$. (The existence of this solution is ensured by the assumptions that are made on the right-hand side of (1) (refer, e.g., to [23])). Similarly, for a real n-vector valued continuous function ψ_0 defined on $[t_0 - \tau, t_0]$ (i.e., ψ_0 is in C_τ), we let $\phi(t, t_0, \psi_0)$, $t \geq t_0 - \tau$, denote the solution of system (4) with the initial condition $\phi(s, t_0, \psi_0) = \psi_0(s)$ for all $s \in [t_0 - \tau, t_0]$. Now if for (1), it is true that $\phi(t, t_0, x_e) = x_e$ for all $t \geq t_0$, then x_e is called an *equilibrium* of system (1). Also, if $\psi_e = x_e$ is a constant function on $[t_0 - \tau, t_0]$ and if for (4), it is true that $\phi(t, t_0, \psi_e) = x_e$ for all $t \geq t_0$, then $x_e = \psi_e$ is called an *equilibrium* of system (4).

The following characterizations pertain to an equilibrium x_e of system (1). Similar statements can be made for system (4).

(i) If it is possible to force the solutions $\phi(t, t_0, x_0)$ to remain as closely as desired to the equilibrium x_e for all $t \geq t_0$ by choosing x_0 sufficiently close to x_e, then the equilibrium x_e is said to be *stable*. If x_e is not stable, then it is said to be *unstable*.

(ii) If an equilibrium x_e is stable and if in addition, the limit of $\phi(t, t_0, x_0)$ as t tends to infinity equals x_e whenever x_0 belongs to $D(x_e)$, where $D(x_e)$ is an open subset of R^n containing x_e, then the equilibrium x_e is said to be *asymptotically stable*. The largest set $D(x_e)$ for which the preceding property is true is called the *domain of attraction* or the *basin of attraction* of x_e. If $D(x_e) = R^n$, then x_e is said to be *asymptotically stable in the large* or *globally asymptotically stable*.

In the case of delay differential equations (4), the preceding definitions are similar (in (ii) above, R^n is replaced by C_τ and $D(x_e)$ is a subset of C_τ).

In the literature on neural networks, the term *global stability* is frequently used. This term should not be confused with the concept of global asymptotic stability introduced above. A neural network (such as, e.g., system (1) or system (4)) is said to be *globally stable* if every trajectory of the system (every solution of the system) converges to some equilibrium. Clearly, a globally stable network cannot sustain oscillations. Usually, global stability in a neural network is established by making use of an energy function or functional for the network in a manner similar to Lyapunov's Second Method, discussed next.

For system (1) (or system (2)), the *Second Method of Lyapunov* (also called the *Direct Method of Lyapunov*) involves the existence of scalar valued functions (defined on R^n) having certain properties. Frequently, these functions, called *Lyapunov functions*, may be viewed as *generalized energy functions* for a given system or as a *generalized distance function* of the system state from a given equilibrium point (which, without loss of generality, may be taken to be the origin). If, for example, such a function is positive definite and if its time derivative along the solutions of system (1) is negative definite, then the equilibrium under discussion (the origin) is *asymptotically stable*. An example of another such result goes (for historical reasons) under the name of *Invariance Principle* and has been used widely in the qualitative analysis of artificial feedback neural networks. When applied, e.g., to system (1), the Invariance Principle has been used to prove that the neural network (1) is *globally stable* if the following conditions are satisfied: (a) there exists a real, scalar valued, and continuous function E defined on R^n such that the time derivative of E along the solutions of system (1), denoted by E', is a non-positive valued function; (b) every solution of system (1) is bounded; and (c) the set of all equilibrium points of system (1) is a discrete set.

For system (4), the Second Method of Lyapunov involves the existence of scalar valued functionals (defined on C_τ), called *Lyapunov functionals*. For such systems, Lyapunov stability results and invariance theorems have been established which are similar to the results cited above for system (1) (or system (2)).

When considering delay differential equations, the above discussion was given for system (4). All of the results enumerated above are applicable to systems with multiple delays as well, such as system (3), by incorporating appropriate refinements which take into account the various time delays τ_k, $k = 1, ..., K$.

For the principal results of the Lyapunov theory (including invariance theorems) for systems described by ordinary differential equations and delay differential equations, refer, e.g., to [8], [22], and [23].

In applying Lyapunov's Second Method to establish global stability for neural networks in the form (1), we usually assume that T is symmetric. When the activation functions $s_i(\cdot)$, $1 \leq i \leq n$, are sigmoidal functions, we utilize an energy function of the form (see, e.g., [10], [11], [20]),

$$E(y) = -\frac{1}{2} y^T T y - U^T y + \sum_{i=1}^{n} b_i \int_0^{y_i} s_i^{-1}(\eta) d\eta \tag{5}$$

where $y = (y_1, ..., y_n)^T$ and $y_i = s_i(x_i)$, $i = 1, ..., n$. When the activation functions are all saturation non-linearities, the energy function associated with system (1) can in this case be chosen as (see, e.g., [3]),

$$E(y) = -\frac{1}{2} y^T T y - U^T y + \frac{1}{2} y^T B y. \tag{6}$$

It is easily verified that for the above choices of E, condition (a) of the Invariance Principle given above is satisfied for system (1). Furthermore, it is easily shown that every solution of system (1) is bounded (see, e.g., [13,14]), and thus, condition (b) of the Invariance Principle is also satisfied for system (1). Finally, it can also be shown that for almost all $U \in R^n$ (except on a set with Lebesgue measure 0), the set of equilibria of system (1) is discrete (see, e.g., [13,14]), and thus, condition (c) of the Invariance Principle, given above, is satisfied as well. This shows that if T

is symmetric, then for almost every $U \in R^n$, system (1) is globally stable.

In the case of system (3) or (4), energy functionals are employed in establishing global stability. We defer our discussion of this topic to the fifth section (dealing with time delay effects).

Globally stable artificial neural networks with discrete sets of equilibria can be used to partition the state space by means of the domains of attraction of the asymptotically stable equilibria of the networks. Such partitions determine equivalence relations which can be employed in a variety of applications of data classification, including associative memories. In such applications, a *desired* set of asymptotically stable equilibria determines a set of neuron output vectors that are used as *stable memories* to store information while the remaining undesirable asymptotically stable equilibria are viewed as *spurious states*. The locations of the desired stable memories are determined by choosing the network interconnections, given by T, in an appropriate manner. We will call this process of selecting T *synthesis*.

Most of the synthesis procedures for associative memories realized by artificial neural networks of the type considered in the present article are motivated by Hebb's hypothesis [20]. In the following, we will summarize three methods that pertain to system (1). We allow the activation functions to be either saturation non-linearities or sigmoidal functions. In the latter case, we assume high gains for the sigmoidal functions (i.e., $\left|\frac{ds_i}{dx_i}(0)\right|$ is very large). We mention in passing that the synthesis methods which we are about to describe are applicable to a variety of artificial feedback neural networks, including the discrete-time Hopfield model (see, e.g., [20]), linear neural networks operating on a hypercube (see, e.g., [14]), and so forth (see, e.g., [1,2,5,9,13]).

The Outer Product Method (see, e.g., [10,11])

We wish to store r desired patterns y^i, $1 \le i \le r$, which correspond to r asymptotically stable equilibria x^i of (1) (i.e., $y^i = S(x^i)$), as stable memories. A set of parameter choices determined by the Outer Product Method are given by

$$T = \sum_{j=1}^{r} y^j (y^j)^T, \quad B = I \text{ and } U = 0 \qquad \text{(OPM)}$$

where I denotes the $n \times n$ identity matrix. The name of this method is motivated by the fact that T consists of the sum of outer products of the patterns that are to be stored as stable memories. This method requires that the y^i, $1 \le i \le r$, be mutually orthogonal (i.e., $(y^i)^T y^j = 0$ when $i \ne j$).

The expression (OPM) constitutes *hard wired learning*. *Adaptive* or *on-line learning* by the outer product method is accomplished by modifying (OPM) as

$$T \to T + \alpha y^l (y^l)^T, \quad B = I, \ U = 0 \qquad (\Delta\text{OPM})$$

where y^l is a new memory to be learned by the network, or by modifying (OPM) as

$$T \to T - \alpha y^l (y^l)^T, \quad B = I, \ U = 0 \qquad (\delta\text{OPM})$$

where y^l is a stored memory which is to be forgotten by the network (see, e.g., [10,11]). In both cases, $\alpha > 0$ is a small constant which determines the rate of learning (or forgetting).

The Projection Learning Rule (see, e.g., [7,25,26])

When the desired prototype patterns $y^j = S(x^j)$, $1 \le j \le r$, to be stored in (1) as stable memories are not mutually orthogonal, a method called the Projection Learning Rule can be used to synthesize the interconnection parameters for (1).

Let

$$\Sigma = [y^1, \ldots, y^r].$$

Recall that for $\Sigma \in R^{r \times n}$, the *Moore-Penrose pseudo-inverse* $\Sigma^I : R^r \to R^n$ defines the linear mapping of any $b \in R^r$ to a unique $x \in R^n$ (i.e., $x = \Sigma^I b$) which has the property that x is the vector that has the smallest Euclidean norm $|x|$ on the set $\{y \in R^n : |\Sigma y - b|^2$ is minimized $\}$. Then the interconnection matrix T for system (1) is given by

$$T = \Sigma \Sigma^I \qquad \text{(PLR)}$$

(refer, e.g., to [20,25,26]). We note that T determined by *(PLR)* satisfies the relation

$$T\Sigma = \Sigma$$

which shows that T is an orthogonal projection of R^n onto the linear space spanned by y^j, $1 \le j \le r$ (hence, the name Projection Rule). It is easily verified that when the y^j, $1 \le j \le r$, are mutually orthogonal, then the Projection Learning Rule and the Outer Product Method coincide. Although the Projection Learning Rule enables us to store *arbitrary* vectors as patterns $y^i = S(x^i)$, $1 \le i \le r$, corresponding to equilibria x^i of (1), there is no guarantee that the equilibria x^i are asymptotically stable.

The (hard wired) learning rule (PLR) can be modified to yield adaptive learning (and forgetting) rules which are in the same spirit as the on-line rules given by (ΔOPM) and (δOPM), since the Moore-Penrose pseudo-inverse can be computed iteratively (see, e.g., [25,26]).

The Eigenstructure Method (see, e.g., [13,14,32])

Neural networks which are synthesized by this method are guaranteed to store desired sets of patterns as stable memories which need not be mutually orthogonal and which correspond to asymptotically stable equilibria of (1). In the following, we assume that the patterns to be stored are bipolar vectors, i.e., $y^j \in B^n = \{y = (y_1, \ldots, y_n)^T \in R^n : y_i = \pm 1, i = 1, \ldots, n\}$, $1 \le j \le r$, and we assume that the activation functions for (1) are saturation non-linearities.

The present method involves a *singular value decomposition* of the $n \times (r - 1)$ matrix

$$Y = [y^1 - y^r, \ldots, y^{r-1} - y^r]$$

given by

$$Y = \Gamma \Sigma W^T$$

where Γ and W are unitary matrices and Σ is a diagonal matrix with the singular values of Y on its diagonal. Letting $\Gamma = [\gamma^1, ..., \gamma^n]$, we recall that $\{\gamma^1, ..., \gamma^n\}$ is an orthonormal basis for the space R^n. If we let k denote the dimension of the linear space L spanned by the vectors $y^1 - y^r, ..., y^{r-1} - y^r$, then $\{\gamma^1, ..., \gamma^k\}$ is an orthonormal basis for L, and $\{\gamma^{k+1}, ..., \gamma^n\}$ is an orthonormal basis of $L^{\perp}$, the orthogonal complement of L. The parameters of the neural network are now given by

$$T = \mu \sum_{i=1}^{k} (\gamma^i)(\gamma^i)^T - \tau \sum_{i=k+1}^{n} (\gamma^i)(\gamma^i)^T \qquad \text{(ESM)}$$

$$U = \mu y^r - T y^r, \quad B = I$$

where I denotes the $n \times n$ identity matrix and $\tau \in R$ and $\mu > 1$ are parameters. It can be shown (see, e.g., [13-16]) that when $\tau > 0$ is sufficiently large, all prototype vectors $y^1, ..., y^r$ are stored as stable memories (corresponding to the asymptotically stable equilibria $x^1, ..., x^r$, where $y^i = S(x^i)$, $i = 1, ..., r$). In fact, all vectors in $L_a \cap B^n$ are stable memories, where L_a is the affine space given by $L + y^r$.

For the eigenstructure method, *iterative learning* (and forgetting) rules which are in the spirit of the adaptive learning rules discussed above for the outer product method and the projection learning rule have also been considered (see, e.g., [32]).

It should be noted that the outer product method and the projection learning rule are motivated by Hebb's hypothesis. Although a relation of the eigenstructure method to Hebbian learning is not obvious, a connection can be established between (ESM) and (OPM) or (PLR) for the case when the stored memories are mutually orthonormal.

For a more detailed and complete discussion of the above synthesis procedures, refer, e.g., to [20].

Effects of Parameter Perturbations

In the implementation process of artificial neural networks, parameter errors are unavoidably encountered. As discussed in the second section (dealing with neural network models), for system (1) such errors include perturbations of (interconnection) weights, $T + \Delta T$; perturbations of the activation functions, $S + \Delta S$; perturbations of the self-feedback terms $B + \Delta B$; and perturbations of the external inputs, $U + \Delta U$. Such inaccuracies will result in errors of the desired stable memories (or corresponding asymptotically stable equilibria), or can even result in the disappearance of stable memories, or the introduction of spurious states. Accordingly, an understanding of qualitative robustness properties of system (1) with respect to parameter variations is of great importance. Of particular interest will be the robust stability of equilibria and error estimates of equilibrium locations.

In the following, we consider neural networks (1) and (2) with activation functions modeled by sigmoidal functions and saturation non-linearities.

Networks with Sigmoidal Activation Functions

In (1) and (2) we assume that the activation functions are sigmoidal functions in the sense described in the second section.

We begin our discussion by making precise the meaning of robustness for system (1). The system (1) is said to be *robust* if for every asymptotically stable equilibrium x_e of (1), there is an asymptotically stable equilibrium $\tilde{x}_e$ of (2) which is near x_e and the distance between x_e and $\tilde{x}_e$, given by $|x_e - \tilde{x}_e|$, can be made as small as desired by requiring that $\max\{|\Delta B|, |\Delta T|, |\Delta U|, |\Delta S(x_e)|, |\Delta S'(x_e)|\}$ be sufficiently small, where $\Delta S'(x) = diag\left[\frac{d(\Delta s_1)(x_1)}{dx_1}, ..., \frac{d(\Delta s_n)(x_n)}{dx_n}\right]$ and where $|\cdot|$ denotes some norm in the real n-vector space R^n or in the $(n \times n)$ matrix space $R^{n \times n}$ (induced by the appropriate corresponding vector norm on R^n). Since all norms in a finite dimensional linear space are equivalent, the above definition is independent of the particular choice of norm $|\cdot|$. (For a more precise ε-δ type definition of robustness of (1), refer to [29]).

Roughly speaking, robustness in the present context means that system (1) is not overly sensitive to small parameter perturbations. In synthesis procedures of associative memories for system (1) (of the type discussed in the third section under background material) robustness ensures that small errors in parameters encountered in the implementation process will not adversely affect the accuracy of the desired stored memories. (That is, robustness ensures that small parameter errors encountered in the implementation process of neural network (1) will not adversely affect the locations of the *desired* asymptotically stable equilibria of system (1) to be used to generate the memories of the network). Clearly, robustness of system (1), as defined above, is of great practical importance in the implementations of such networks.

A natural question to ask is whether system (1) is perhaps always robust. To see that this is not the case, consider the scalar equation

$$\dot{x} = -\frac{2}{\pi} x + \frac{2}{\pi} \arctan x \qquad (7)$$

which is clearly a special case of system (1). For (7), $x_e = 0$ is an asymptotically stable equilibrium. However, for any fixed $\varepsilon > 0$, the perturbation of system (7) given by

$$\dot{x} = -\frac{2}{\pi} x + (1 + \varepsilon)\frac{2}{\pi} \arctan x \qquad (8)$$

has no asymptotically stable equilibrium in a sufficiently small neighborhood of $x_e = 0$ (refer to [29] for details).

It is shown in [29] that a necessary and sufficient condition for the robustness of system (1) (in the sense defined above) is that for every asymptotically stable equilibrium x_e of system (1), x_e is asymptotically stable with respect to the linearization of system (1) (near x_e). This condition can be verified by testing the Hurwitz stability of the coefficient matrix

$$-B + TS'(x_e)$$

for each asymptotically stable equilibrium x_e (recall that a matrix is Hurwitz stable if all its eigenvalues have negative real parts).

When the above condition is satisfied (i.e., when system (1) is robust), Brouwer's fixed point theorem (see [28]) can be used to obtain the following estimate of the distance between the equilibrium x_e of (1) and the corresponding perturbed equilibrium $\tilde{x}_e$ (of system (2)),

$$|\tilde{x}_e - x_e|_\infty \le c \max\{|\Delta B|_\infty, |\Delta T|_\infty, |\Delta S(x_e)|_\infty, |\Delta U|_\infty\} \qquad (9)$$

when in the right-hand side of inequality (9), the maximal number is sufficiently small. In the above, $|\cdot|_\infty$ denotes either the infinity vector norm or the matrix norm induced by the infinity vector norm. Also, $c = 2(2 + R_0 + |T|_\infty)|A^{-1}|_\infty$ with $R_0 \ge |x_e|_\infty$ and $A = -B + TS'(x_e)$.

A more complete statement of the above result is as follows. Assume that x_e is an asymptotically stable equilibrium with respect to the linearization of system (1) near x_e (i.e., the matrix $A = -B + TS'(x_e)$ is Hurwitz stable which implies that x_e is an asymptotically stable equilibrium with respect to system (1)). Then there is a constant $M > 0$ (which can be expressed explicitly—refer to [29]) such that if

$$\max\{|\Delta B|_\infty, |\Delta T|_\infty, |\Delta S(x_e)|_\infty, |\Delta S'(x_e)|_\infty, |\Delta U|_\infty\} < M,$$

then there exists a vector $\tilde{x}_e$ which is an asymptotically stable equilibrium of the perturbed system (2) and $\tilde{x}_e$ satisfies the estimate (9).

Summarizing, when the neural network (1) is robust and when the implementation errors for (1) are reasonably small, then

(i) for every desired memory there exists a corresponding actual stored memory; and

(ii) errors in memories ≤ $c\times$ (parameter errors in the implemented network), where c is a computable constant (as specified earlier).

Networks with Saturation Activation Functions

In the following, we assume that in (1) and (2) the activation functions are saturation non-linearities (i.e., $y = (y_1, ..., y_n)^T = (s_1(x_1), ..., s_n(x_n))^T = (sat(x_1), ..., sat(x_n))^T \overset{\Delta}{=} sat(x)$), that $\Delta S(x) = 0$, and that only bipolar vectors (i.e., vectors belonging to the set B^n) are to be considered as candidates for desired stable memories.

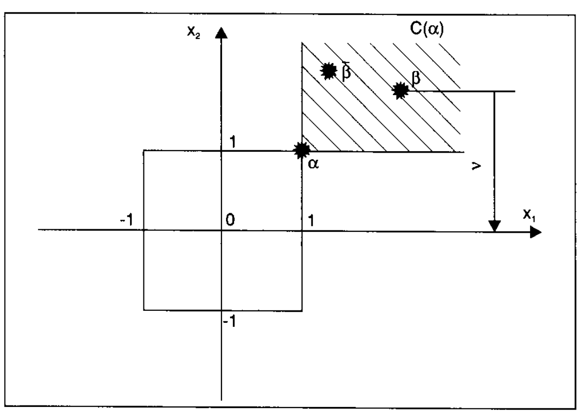


Fig. 1. A geometric interpretation of the robustness analysis.

For x in R^n, we let $\delta(x) = \min_{1\le i\le n}\{|x_i|\}$ and for $\alpha = (\alpha_1, ..., \alpha_n)^T \in B^n$ we let $C(\alpha) = \{x \in R^n : x_i\alpha_i > 1\}$ (refer to Fig. 1). Now suppose that $\alpha^1, ..., \alpha^r \in B^n$ are desired stable memories of system (1) corresponding to the asymptotically stable equilibria $\beta^1, ..., \beta^r$, respectively (i.e., $\alpha^j = sat(\beta^j)$, $j = 1, ..., r$). Let

$$v = \min_{1\le j\le r}\{\delta(\beta^j)\} > 1. \qquad (10)$$

It is shown in [16] that $\alpha^1, ..., \alpha^r$ are also stable memory vectors of system (2) provided that

$$|B^{-1}(\Delta B)|_\infty + |B^{-1}(\Delta T)|_\infty + |B^{-1}(\Delta U)|_\infty < v - 1 \qquad (11)$$

This robustness criterion is proved by using the result (which is proved in [15]) that if $\alpha \in B^n$ and if $\beta = B^{-1}(T\alpha + U) \in C(\alpha)$, then α is a stable memory and β is a corresponding asymptotically stable equilibrium (i.e., $\alpha = sat(\beta)$) for system (1).

Now suppose that α is a stable memory and β is a corresponding asymptotically stable equilibrium of system (1). After perturbation, the new asymptotically stable equilibrium point $\bar{\beta}$ is given by

$$\bar{\beta} = (B + \Delta B)^{-1}[(T + \Delta T)\alpha + (U + \Delta U)]. \qquad (12)$$

When condition (11) is satisfied, it can be shown that $\bar{\beta} \in C(\alpha)$, which implies that α is still a stable memory (of (2)).

To give a geometric interpretation of the above, assume that $\alpha \in R^2$ is a desired stable memory of system (1) with corresponding asymptotically stable equilibrium of (1) given by β. Then $\beta = B^{-1}(T\alpha + U)$ must be in the region $C(\alpha)$, since $v = \min\{\delta(\beta)\} > 1$ (refer to Fig. 1).

The perturbations ΔB, ΔT, and ΔU give rise to a displacement of the equilibrium β (of system (1)) to a new equilibrium $\bar{\beta}$ (of (2)). In order for α to remain an invariant stable memory for system (1) after perturbation (i.e., in order for α to be a stable memory for system (2)), we require that $\bar{\beta}$ also be in $C(\alpha)$. It is clear that as long as $\bar{\beta}$ remains in $C(\alpha)$, α will be a stable memory of the perturbed system (2). This robustness result provides one of the *possible* upper bounds for the perturbations, specified by $|B^{-1}(\Delta B)|_\infty + |B^{-1}(\Delta T)|_\infty + |B^{-1}(\Delta U)|_\infty$, which will ensure that the perturbed equilibrium $\bar{\beta}$ and the original equilibrium β are within the same region $C(\alpha)$. This upper bound is given by $v - 1$ where v is given in (10).

For system (2) we require that $b_i + \Delta b_i > 0$ for each i. It is clear that a perturbation ΔB with $\Delta b_i < 0, i = 1, ..., n$, will not change the desired memory vectors $\alpha^1, ..., \alpha^r \in B^n$ of system (1) (refer to (12)).

When considering perturbations due to an implementation process, the focus is usually on the interconnection matrix T and not on the parameters B and U. When the latter can be ignored (i.e., when we can assume $\Delta B = 0$, $\Delta U = 0$), then condition (11) assumes the simple form

$$|B^{-1}(\Delta T)|_{\infty} < \nu - 1. \quad (13)$$

In closing, we observe that the concept of robustness introduced earlier in the first subsection of the present section (for system (1) with sigmoidal activation functions) is applicable to the present discussion as well (for system (1) with saturation non-linearities). We conclude from (11) that under the present assumptions (that $\Delta S = 0$ and that the desired stored memories be bipolar vectors), system (1) will always be robust (in the sense of the present context). Not surprisingly, this tells us that for applications of the type considered herein (e.g., associative memories with bipolar memory vectors), system (1) with saturation non-linearities for activation functions will in general be less sensitive with respect to parameter perturbations than system (1) with sigmoidal non-linearities for activation functions.

Effects of Time Delays

In the implementation process of artificial feedback neural networks, especially by VLSI, transmission delays are unavoidably introduced. It is known that in globally stable feedback neural networks without time delays, oscillations can occur after the introduction of delays (see, e.g., [9] and [17]). It is therefore important to take the effects of time delays into account in the qualitative analysis of such networks.

To simplify our discussion, we present global and local results for system (4) endowed with the same time delay for various state variables. We then present extensions of these results to systems with multiple delays described by equation (3). In both cases, these results show that for a given set of parameters system (1) and system (3) (or system (4)) will possess similar global and local qualitative behavior, provided that the time delays are sufficiently small.

Global Stability Results

We first assume that in system (4) the activation functions s_i are sigmoidal functions and we assume that the matrix $T = T_1 + T_2$ is symmetric. In applying the Invariance Principle discussed in the third section (dealing with background material) to establish global stability of system (4), we associate with this system the energy functional given by

$$E(x_t) = -\frac{1}{2} y^T(t)Ty(t) - U^T y(t) + \sum_{i=1}^{n} b_i \int_0^{y_i(t)} s_i^{-1}(\sigma)d\sigma$$
$$+ \int_{t-\tau}^{t} [y(w) - y(t)]^T T_2^T f(w-t) T_2 [y(w) - y(t)]dw \quad (14)$$

where $x_t(\cdot)$ denotes a continuous function on $[-\tau, 0]$ defined by $x_t(s) = x(t+s)$ for all $s \in [-\tau, 0]$, $y(t) \stackrel{\Delta}{=} S(x(t))$, and f is a continuously differentiable, non-negative, scalar valued function defined on $[-\tau, 0]$. It can be shown that there exists a function $f(\cdot)$ such that the time derivative of E along the solutions of system (4) is a non-positive valued function. The determination of such an $f(\cdot)$ is lengthy. We refer the interested reader to [30] for the details of these computations. It turns out that a sufficient condition for the existence of such $f(\cdot)$ is given by

$$\tau\beta|T_2|_2 < 1 \quad (15)$$

where $\tau > 0$ denotes the time delay in system (4),

$$\beta \stackrel{\Delta}{=} \max_{x \in R^n} |S'(x)|_2, \; S'(x) = diag\left[\frac{ds_1}{dx_1}(x_1), \ldots, \frac{ds_n}{dx_n}(x_n)\right],$$

and $|\cdot|_2$ denotes the matrix norm induced by the Euclidean vector norm (i.e., $|A|_2 = [\lambda_{max}(A^T A)]^{1/2}$, where $\lambda_{max}(A^T A)$ is the largest eigenvalue of $A^T A$).

Now similarly as in the case of system (1) (without delays), it is easily shown that every solution of system (4) is bounded and for almost all $U \in R^n$ (except a set with Lebesgue measure zero), the set of equilibria of system (4) is a discrete set (refer to [30]). It now follows from the Invariance Principle that when $T = T_1 + T_2$ is symmetric and when (15) is satisfied for almost every $U \in R^n$, then system (4) is globally stable.

To get an idea of how conservative (15) *might* be, we refer the reader to [17], where through a linearization process, the bound $\tau\beta\lambda_{min}(T) < \pi/2$ for the (*local*) asymptotic stability of an equilibrium of a special case of (4) is obtained. Backed by a simulation study, it is then *conjectured* in [17] that if the above bound is true at all asymptotically stable equilibria of (4), then (4) is globally stable.

For the case when all components of $S(x)$ in (4) are saturation non-linearities, an energy functional which is similar to (14) (with the term $\sum_{i=1}^{n} b_i \int_0^{y_i(t)} s_i^{-1}(\sigma)d\sigma$ replaced by the term $\frac{1}{2}\sum_{i=1}^{n} b_i[y_i(t)]^2$) is used in the application of the Invariance Principle to obtain (15) as a sufficient condition for the global stability of system (4) for almost all $U \in R^n$ when $T = T_1 + T_2$ is symmetric. (Under additional assumptions on T, a more conservative result for the global stability of (4), given by $\tau\beta|T_2|_2 < 2/3$, was originally established in [4].)

Using a refinement to the functional given in (14) involving the various delays for system (3), the Invariance Principle is invoked in [31] to establish the sufficient condition for the global stability of (3) given by

$$\sum_{k=1}^{K} \tau_k |T_k|_2 \beta < 1 \quad (16)$$

where $T_k = \left[t_{ij}^{(k)}\right]_{n \times n}$ (refer to (3)), and β is the same as in inequality (15).

Local Stability Results

As mentioned earlier, in the application of artificial neural networks to associate memories, as well as in other applications, the aim is to store information in stable memories which correspond to specific asymptotically stable equilibria. Good criteria which ensure such (local) stability properties are therefore very important. We address this question in the following for system (3) and (4).

We begin by emphasizing that even for the case of *linear* delay equations given by

$$\dot{x}(t) = Ax(t) + Bx(t - \tau) \qquad (17)$$

where A and B are real $n \times n$ matrices and $\tau > 0$ is a delay, there are no known general results which constitute necessary and sufficient conditions for the asymptotic stability of the equilibrium $x_e = 0$ (see, e.g., [8]). Accordingly, the problem of determining the local stability properties of an equilibrium of a delay differential equation is non-trivial, even in the case of linear systems. It turns out, however, that by utilizing the special structure of Hopfield neural networks with time delays, we can prove that under the conditions of global stability given in the preceding subsection, the asymptotic stability of any equilibrium of system (4) (or of system (3)) can be deduced from the asymptotic stability of the same corresponding equilibrium of system (1). In other words, if (15) is satisfied for system (4) (or if (16) is satisfied for system (3)), then the Hopfield neural network (1) without delays and the Hopfield neural network with delay, (4) (or the Hopfield neural network with multi-delays, (3)) are globally stable (for almost every $U \in R^n$), and furthermore, both system (1) and system (4) (or, both system (1) and system (3)) have identical equilibria with the same local stability properties at each equilibrium.

We summarize the above results in the following *more general* and precise equivalent statements (given here for system (4)). When system (4) satisfies (15) (and thus, (4) is globally stable), then the following statements are equivalent (refer to [30] for the proof):

(i) x_e is a stable equilibrium of (4);
(ii) x_e is an asymptotically stable equilibrium of (4);
(iii) x_e is a local minimum of the energy functional given by (14); •
(iv) x_e is a stable equilibrium of (1);
(v) x_e is an asymptotically stable equilibrium of (1); and
(vi) x_e is a local minimum of the energy function E given by (5).

We now see from the above results that when the time delay τ is sufficiently small (i.e., when $\tau\beta|T_2|_2 < 1$), then a study of the stability properties of the equilibria of a Hopfield neural network with delay (system (4)) can be reduced to a study of the stability properties of the equilibria of a corresponding Hopfield neural network without delays (system (1)).

All of the preceding statements apply also to Hopfield neural networks with multi-delays (system (3)) with condition (15) replaced by condition (16) (refer to [31]).

For specific examples which demonstrate applications of the results given in the present section, refer to [17,18,30,31].

We conclude the present section by noting that the robustness properties with respect to time delays discussed above are not only true for Hopfield-like neural networks, but for a much broader class of recurrent artificial neural networks, such as, e.g., Cohen-Grossberg neural nets (refer to [2] and [5]).

Interconnection Constraints

One of the major difficulties encountered in VLSI implementations of artificial neural networks is the realization of extremely large numbers of interconnections in the network. Current VLSI technology restricts the connectivity in neural network implementations to a level at which one cannot expect to achieve more than a few hundred neurons in an implemented neural network chip. Accordingly, in many applications where large numbers of neurons are required, fully connected neural networks are not practical. This has motivated a specific neural network structure, the *cellular neural network structure* (see, e.g., [3]), in which one considers only *local* interconnections (among neurons), restricted to small neighborhoods.

In this section we present synthesis procedures for associative memories involving neural networks with three different interconnection constraints. The methodology advanced herein is potentially also applicable to the Outer Product Method and the Projection Learning Rule (summarized in the third section), as well as other synthesis procedures. For purposes of discussion, we confine ourselves in the following to a procedure which constitutes a generalization of the Eigenstructure Method presented in the third section and which incorporates the perturbation and robustness results of the fourth section for the case of system (1) with saturation activation functions (refer to the second subsection of the fourth section). We will consider three specific cases: no interconnection constraints, prespecified sparsity constraints with no symmetry requirement, and prespecified sparsity constraints with symmetric interconnection requirement.

Networks Without Interconnection Constraints

Suppose we are given a set of desired patterns $\alpha^1, ..., \alpha^r$. We wish to synthesize a system of form (1) which stores $\alpha^1, ..., \alpha^r$ as stable memories. Without loss of generality, we choose B as the $n \times n$ identity matrix and choose $\beta^j = \mu\alpha^j$, for $j = 1, ..., r$, with $\mu > 1$ (hence, $\beta^j \in C(\alpha^j)$). Then T and U will be determined by the relations

$$B\beta^j = \mu\alpha^j = T\alpha^j + U, j = 1, ..., r. \qquad (18)$$

Solutions of (18) for T and U will always exist. To see this, let $Y = [\alpha^1 - \alpha^r, ..., \alpha^{r-1} - \alpha^r]$. We need to solve T from

$$TY = \mu Y \qquad (19)$$

and set

$$U = \mu\alpha^r - T\alpha^r. \qquad (20)$$

This guarantees that system (1) will store the desired patterns α^1, ..., α^r as stable memories and $\beta^1, ..., \beta^r$ as corresponding asymptotically stable equilibrium points. Solutions of (19) (for T) always exist since

$$rank[Y] = rank\begin{bmatrix} Y \\ \cdots \\ \mu Y \end{bmatrix}$$

Indeed, a trivial solution for T is the $n \quad n$ identity matrix multiplied by μ. We wish to determine non-trivial solutions for matrix T. This can be accomplished by many methods, including the *singular value decomposition method* (refer to the third section). Performing a singular value decomposition of Y, we obtain

$$Y = [\Gamma_1 \vdots \Gamma_2] \begin{bmatrix} D & \vdots & 0 \\ \cdots & \vdots & \cdots \\ 0 & \vdots & 0 \end{bmatrix} \begin{bmatrix} V_1^T \\ \cdots \\ V_2^T \end{bmatrix} \tag{21}$$

where $D \in R^{p \times p}$ is a diagonal matrix with the non-zero singular values of matrix Y on its diagonal and

$$p = rank\ [Y]. \tag{22}$$

Then, (19) yields

$$T\Gamma_1 = \mu\Gamma_1. \tag{23}$$

Solutions of (23) for T can be expressed as

$$T = \mu\Gamma_1\Gamma_1^T + W\Gamma_2^T \tag{24}$$

where W is an arbitrary $n \times (n - p)$ real matrix. It can easily be verified that T given in (24) is also a solution of (19) and it is non-trivial when $p < n$. The bias vector U can be computed as in (20). We note that by special choice of matrix W, (24) can result in a symmetric matrix T (refer to the third section and to [14] for a special case).

The above steps constitute a synthesis procedure of neural networks with *no constraints* on the interconnecting structure. This procedure usually results in a *fully interconnected* neural network. The consequence of the above design is that $\alpha^1, ..., \alpha^r$ will be stored as stable memory vectors in system (1), that the states β^j corresponding to α^j, $j = 1, ..., r$ will be asymptotically stable equilibrium points of system (1), and that all vectors in $L\alpha \cap B^n$ including $\alpha^1, ..., \alpha^r$, will be stored as memory vectors in system (1), where $L\alpha = \text{Aspan}(\alpha^1, ..., \alpha^r) \overset{\Delta}{=} \text{Span}(\alpha^1 - \alpha^r, ..., \alpha^{r-1} - \alpha^r) + \alpha^r$ and Span $(x_1, ..., x_n)$ denotes the linear subspace of R^n generated by $x_1, ..., x_n$.

Networks with Prespecified Sparsity Constraints

The synthesis technique developed above will result in neural networks with *symmetric or non-symmetric* coefficient matrix T which in general will not be sparse. Fully interconnected artificial neural networks with even a moderate number of neurons give rise to large numbers of *line-crossings* resulting from the network interconnections, and thus, pose formidable obstacles in VLSI implementations. For this reason, it is desirable to establish synthesis procedures which will result in interconnecting structures which do not demand large numbers of interconnections. We will first consider the procedure which does not require symmetric interconnections.

Sparsity constraints on the interconnecting structure for a given neural network are usually expressed as constraints which require that predetermined elements of T be zero. To simplify the subsequent discussion, we consider without any loss of generality the specific case when $n = 4$ and the constraints on T are given, for example, by

$$T = \begin{bmatrix} T_{11} & 0 & T_{13} & 0 \\ 0 & T_{22} & 0 & T_{24} \\ T_{31} & 0 & T_{33} & 0 \\ 0 & T_{42} & 0 & T_{44} \end{bmatrix}, \tag{25}$$

where the T_{ij}'s are to be determined. The question to be answered is whether for a given $4 \times (r - 1)$ matrix Y, it is possible to determine (non-trivial) solutions of T with structure (25) from the matrix equation (19). We will show in the following that (non-trivial) solutions for such T always exist as long as all the diagonal elements of matrix T are assumed to be non-prespecified elements (e.g., as given in (25)) and $p < n$ (p is defined in (22)). One class of sparsely interconnected neural networks which satisfies the above structural condition are *cellular neural networks*, [3]. Cellular neural networks (which are also described by equation (1)), require that the matrix T have a special *sparse* structure in which all the diagonal elements of T are required to be non-zero.

Solutions of Equation (19) for T with prespecified zero entries will *always* exist, provided that the conditions mentioned above are satisfied. To see this. we write (19) as

$$T_iY = \mu Y_i \text{ for } i = 1, ..., n, \tag{26}$$

where T_i and Y_i represent the ith row of T and Y, respectively. For the example considered in (25), we have, when $i = 2$,

$$[0 \quad T_{22} \quad 0 \quad T_{24}]Y = [0 \quad T_{22} \quad 0 \quad T_{24}] \begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \end{bmatrix} = \mu Y_2.$$

This equation is equivalent to

$$[T_{22} \quad T_{24}] \begin{bmatrix} Y_2 \\ Y_4 \end{bmatrix} = \mu Y_2, \tag{27}$$

and solutions of (27) for $[T_{22}\ T_{24}]$ always exist since

$$rank \begin{bmatrix} Y_2 \\ Y_4 \end{bmatrix} = rank \begin{bmatrix} Y_2 \\ Y_4 \\ \mu Y_2 \end{bmatrix}. \tag{28}$$

Generally speaking, when T_{ii}, $i = 1, ..., n$, are not prespecified as zero elements, one of the rows of Y appearing on the right-hand side of (27) will also appear on the left-hand side, which implies that the rank condition (28) is satisfied. The condition $p < n$ is also required, since when $p = n$, one cannot guarantee to find a non-trivial solution ($p = n$ will sometimes result in the trivial solution T equal to the identity matrix [14]). Solutions of (27) can be determined using the singular value decomposition method as was done when solving (19). Specifically, we perform a singular value decomposition of

$$\begin{bmatrix} Y_2 \\ Y_4 \end{bmatrix} = [\Gamma_{i1} \vdots \Gamma_{i2}] \begin{bmatrix} D_i & \vdots & 0 \\ \cdots & \vdots & \cdots \\ 0 & \vdots & 0 \end{bmatrix} \begin{bmatrix} V_{i1}^T \\ \cdots \\ V_{i2}^T \end{bmatrix}, \tag{29}$$

and determine

$$[T_{22} \quad T_{24}] = \mu Y_2 V_{i1} D_i^{-1} \Gamma_{i1}^T + W_i \Gamma_{i2}^T, \tag{30}$$

where W_i is an arbitrary row vector with appropriate dimension and $i = 2$ for the example in (27). The bias vector $U = [U_1, ..., U_n]^T$ is computed in a similar manner as in (20). The consequence of this synthesis procedure is that $\alpha^1, ..., \alpha^r$ will be stored as stable memory vectors for system (1) with B, T, and U determined as above, that the states β^i corresponding to α^i, $i = 1, ..., r$, will be asymptotically stable equilibrium points of the synthesized system, and that the matrix T has the desired *sparse* interconnecting structure.

It is readily seen that the synthesis procedures presented above guarantee that $\alpha^1, ..., \alpha^r$ are also memory vectors of system (2) provided that (refer to equation (11))

$$|B^{-1}\Delta B|_\infty + |B^{-1}\Delta T|_\infty + |B^{-1}\Delta U|_\infty = |\Delta B|_\infty + |\Delta T|_\infty + |\Delta U|_\infty < \mu - 1. \tag{31}$$

The above enables us to specify *an upper bound* for the parameter inaccuracies encountered in the implementation of a given network design to store a desired set of bipolar patterns in system (1). This bound is chosen by the designer during the initial phase of the design procedure. This type of flexibility does not appear to have been achieved before (e.g., [6,7,10,11,13,14,20,21,24-27]. Specifically, the synthesis procedure advocated above incorporates two features which are very important in the VLSI implementation of artificial neural networks: (i) it allows the VLSI designer to choose a suitable interconnecting structure for the neural network; and (ii) it takes into account inaccuracies which arise in the realization of the neural network by hardware.

Networks with Sparsity and Symmetry Constraints

In this subsection, a synthesis procedure for associative memories is presented which results in *sparse and symmetric* interconnection matrices T for system (1). Presently, we assume that in (2) $\Delta S = 0$, $\Delta B = 0$, and $\Delta U = 0$.

For the B, T, and U determined by the synthesis procedure with sparsity constraints with $\mu > 1$, we choose

$$\Delta T = (T^T - T)/2. \tag{32}$$

Then, $T_s \overset{\Delta}{=} T + \Delta T = (T + T^T)/2$ is a symmetric matrix. From our robustness analysis result, we note that if

$$|B^{-1}\Delta T|_\infty = |T^T - T|_\infty/2 < \mu - 1, \tag{33}$$

the neural network (2) will also store all the desired patterns as memories, with a symmetric connection matrix $T + \Delta T = T_s$.

The above observation gives rise to the possibility of designing a neural network (1) with *prespecified interconnection structure* and with a *symmetric interconnection matrix*. Such capability is of *great* interest since neural network (1) will be *globally stable* when T is symmetric [20]. It appears that (33) might be satisfied by choosing μ sufficiently large. However, from (30) it is seen that large μ will usually result in large absolute values of the components of T which in turn may result in a large $|T^T - T|_\infty$. Therefore, it is not always possible for (33) to be satisfied by choosing μ large. From (33), we see that if our original synthesized matrix T is sufficiently close to its symmetric part $(T + T^T)/2$, or equivalently, if $|T^T - T|_\infty$ is sufficiently small, then (33) is satisfied and we are able to design a neural network of form (1) with the following properties: (i) the network stores $\alpha^1, ..., \alpha^r$ as stable memory vectors; (ii) the network has a predetermined (full or sparse) interconnecting structure; and (iii) the connection matrix T of the network is symmetric.

An *iterative algorithm* (design procedure) can be utilized to achieve this. Let ΔT be defined as in (32). For the given $\mu > 1$, suppose that $|\Delta T|_\infty \geq \mu - 1$. We can find a λ, $0 < \lambda < 1$, such that $\lambda|\Delta T|_\infty < \mu - 1$, and we let $T_1 = T + \lambda\Delta T$. We use this T_1 as the *new* connection matrix for our neural network (1). According to our robustness analysis, we see that $\alpha^1, ..., \alpha^r$ are still memory vectors of system (1) with coefficient matrix T_1, and we can compute the corresponding asymptotically stable equilibrium points as $\bar{\beta}^j = B^{-1}(T_1\alpha^j + U)$ for $j = 1, ..., r$. Clearly $\bar{\beta}^j \in C(\alpha^j)$. We can determine the upper bound v for the permissible perturbation ΔT as in (10) and (11), where we use $\bar{\beta}^j$ instead of β^j. We repeat the above procedure, until we determine a symmetric coefficient matrix T or until we arrive at $v \leq 1 + \eta$ (where η is a small positive number, e.g., $\eta = 0.001$).

If we end up with $T = T^T$, we have found a solution. If we end up with $v \leq 1 + \eta$ and $T \neq T^T$, our design procedure is not successful in solving a symmetric T for the given problem. Experimental results indicate that this procedure will frequently succeed in determining a symmetric matrix T [16].

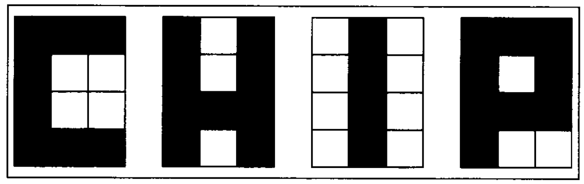


Fig. 2. The four desired memory patterns.

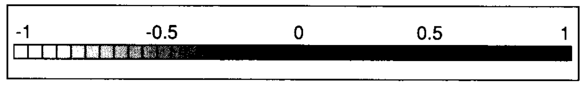


Fig. 3. Grey levels.

Examples

We now apply the results of the preceding section to specific cases. In all of these, we consider a neural network with 12 neurons ($n = 12$) and our objective is to store the four patterns shown in Fig. 2 as memories. As indicated in this figure, twelve boxes are used to represent each pattern (in R^{12}), with each box corresponding to a vector component which is allowed to assume values between -1 and 1. For the purpose of visualization, -1 will

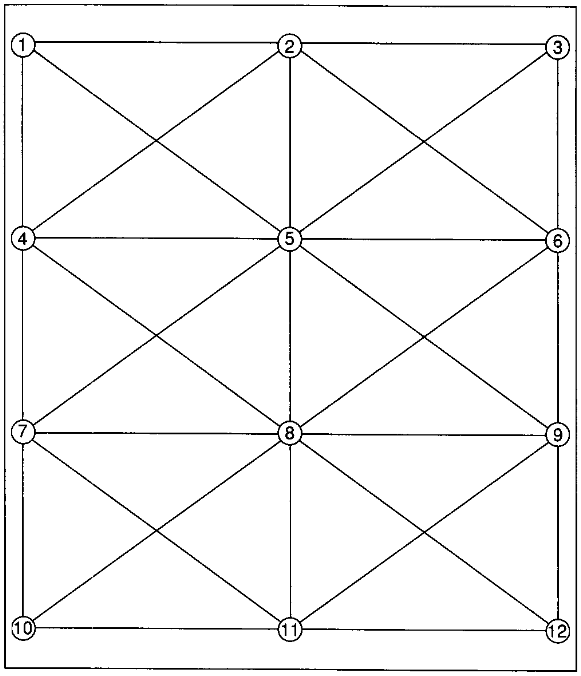


Fig. 4. Interconnecting structure of a cellular neural network.

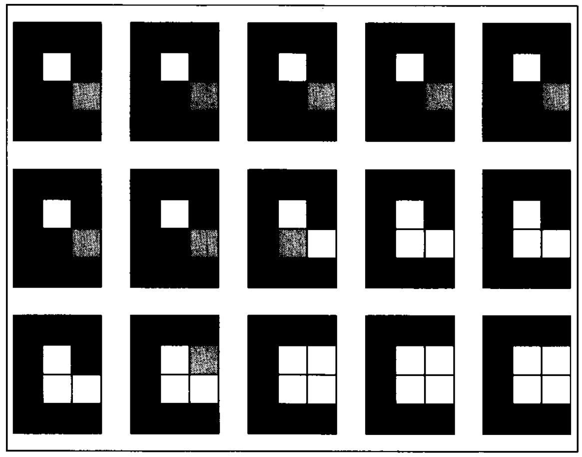


Fig. 5. A typical evolution of pattern no. 1 of Fig. 2.

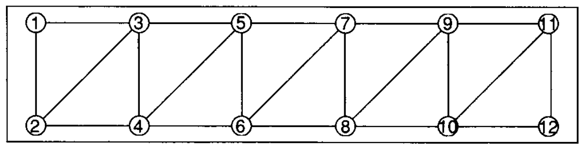


Fig. 6. A possible structure for a neural network without line-crossings in the interconnecting structure.

represent white, 1 will represent black, and the intermediate values will correspond to appropriate grey levels, as shown in Fig. 3. The four desired patterns in Fig. 2 correspond to the following four bipolar vectors:

$\alpha^1 = [1,1,1,1,-1,-1,1,-1,-1,1,1,1]^T, \alpha^2 = [1,-1,1,1,-1,1,1,1,1,1,-1,1]^T$

$\alpha^3 = [-1,1,-1,-1,1ß^T, \alpha^4 = [1,1,1,1,-1,1,1,1,1,1,-1,-1]^T.$

The cases which we consider below involve different prespecified constraints on the interconnecting structure of each network.

Case I: Cellular neural network. We synthesized a cellular neural network with the configuration given in Fig. 4. In doing so, we chose B as the 12 12 identity matrix, $\mu = 15$, and we determined T and U as specified in Table I.

The performance of this network is illustrated by means of a typical simulation run of equation (1), shown in Fig. 5. In this figure, the desired memory pattern is depicted in the lower right corner. The initial state, shown in the upper left corner, is generated by adding to the desired pattern zero-mean Gaussian noise with a standard deviation SD=1. The iteration of the simulation evolves

$T =$

3.3333e−01	−1.0991e−15	0	3.3333e−01	−1.4333e+01	0	0	0	0	0	0	0
−3.5000e+00	1.5000e+01	−3.5000e+00	−3.5000e+00	−1.0500e+01	0	0	0	0	0	0	0
0	0	5.0000e−01	0	−1.4500e+01	1.7764e−15	0	0	0	0	0	0
2.5000e−01	0	0	2.5000e−01	−1.4250e+01	0	2.5000e−01	0	0	0	0	0
−5.9412e+00	−8.3438e−16	−5.9412e+00	−5.9412e+00	−8.0588e+00	3.5294e−01	−5.9412e+00	−7.0588e−01	3.5294e−01	0	0	0
0	3.1371e−15	−5.1250e+00	0	−8.8750e+00	5.6250e+00	0	3.7500e+00	5.6250e+00	0	0	0
0	0	0	−2.1111e+00	−1.1889e+01	0	−2.1111e+00	−9.4444e+00	0	−2.1111e+00	−9.4444e+00	0
0	0	0	−7.1579e+00	−6.8421e+00	2.1053e−01	−7.1579e+00	3.6842e−01	2.1053e−01	−7.1579e+00	−1.4211e+01	−3.8186e−16
0	0	0	0	−3.1429e+00	−7.1429e−01	0	3.1429e+00	−7.1429e−01	0	−1.3286e+01	5.4089e−16
0	0	0	0	0	0	1.8000e+00	−1.1400e+01	0	1.8000e+00	−1.1400e+01	0
0	0	0	0	0	0	−5.3750e+00	−1.0750e+01	−9.1250e+00	−5.3750e+00	−4.8750e+00	−4.3001e−16
0	0	0	0	0	0	0	2.1089e−16	−7.0000e+00	0	−7.0000e+00	1.5000e+01

$$U = [0,\ 1.0658e-14,\ -1.7764e-15,\ -7.1054e-15,\ 7.0588e-01,\ -3.7500e+00,\ 9.4444e+00$$
$$1.4632e+01,\ -3.1429e+00,\ 1.1400e+01,\ 1.0750e+01,\ 8.8818e-15]^T$$

Table I.

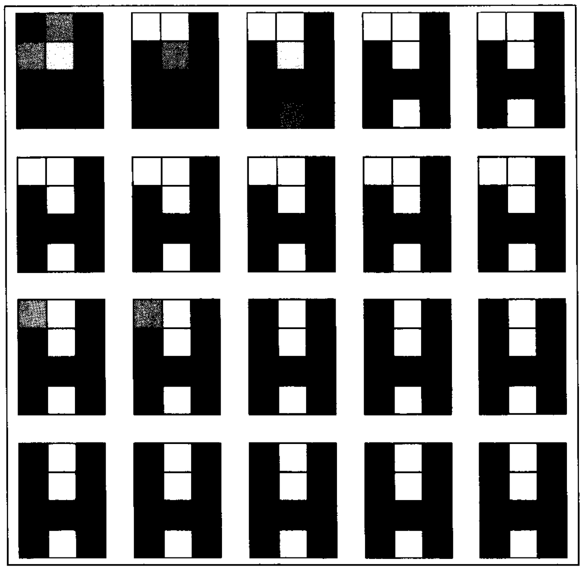


Fig. 7. A typical evolution of pattern no. 2 of Fig. 2.

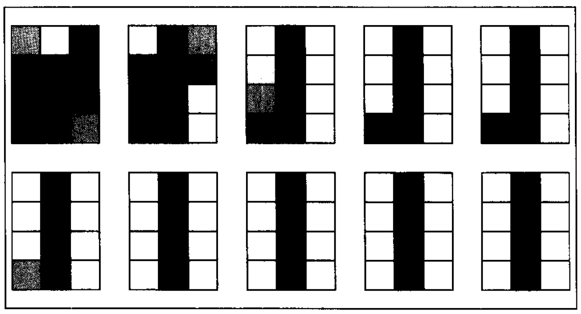


Fig. 8. A typical evolution of pattern no. 3 of Fig. 2.

from left to right in each row and from the top row to the bottom row. The desired pattern is recovered in 12 steps with a step size h = 0.06 in the simulation of equation (1). We do not identify a unit for the step size h. All simulations for the present examples were performed on a Sun SPARC Station using MATLAB.

Case II: Quinquediagonal interconnecting structure resulting in no line-crossings. We chose the interconnecting structure shown in Fig. 6 in which there are no line-crossings, resulting in a quinquediagonal matrix T. We note that this configuration can be generalized to arbitrary n. A typical simulation run for the present case is depicted in Fig. 7. In this figure, the noisy pattern was generated by adding Gaussian noise $N(0.1, 0.7)$ to the desired pattern. Convergence occurred in 16 steps with $h = 0.06$.

Case III: Non-symmetric T with perturbations. We generated randomly a matrix ΔT, given in Table II, which has the same sparse structure as T and which satisfies the condition that $|\Delta T|_\infty < \mu - 1$, where T was obtained from Case I. We used $T_3 \stackrel{\Delta}{=} T + \Delta T$ in system (2) (with $\Delta B = 0$, $\Delta U = 0$).

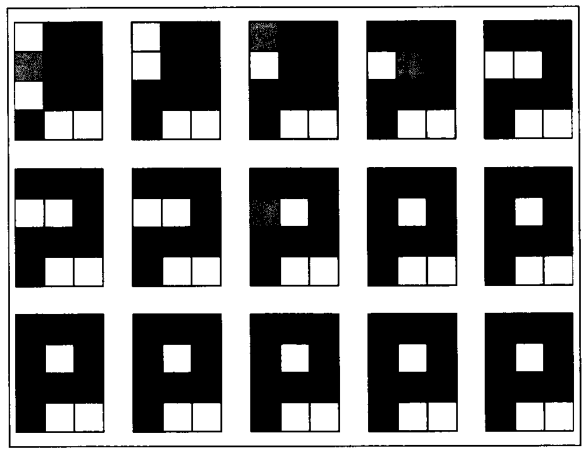


Fig. 9. A typical evolution of pattern no. 4 of Fig. 2.

$$\Delta T = \begin{bmatrix} -1.0794 & -1.1747 & 0 & 0.8750 & 0.3025 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1.1603 & 0.6712 & -0.8235 & -1.6491 & -0.1272 & 1.1939 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.3633 & -0.0644 & 0 & 1.7388 & 1.1622 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.1312 & -1.6710 & 0 & 1.8394 & -0.1047 & 0 & -1.2227 & 0.1839 & 0 & 0 & 0 & 0 \\ 1.6731 & -0.9829 & -0.9946 & -1.4540 & 0.9143 & 1.8469 & -1.4393 & -0.5019 & -1.5160 & 0 & 0 & 0 \\ 0 & 0.0208 & 0.3095 & 0 & -1.8896 & 1.6014 & 0 & 1.3049 & 1.1495 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.6675 & 1.0976 & 0 & -0.8437 & -1.2986 & 0 & -1.2319 & -0.9470 & 0 \\ 0 & 0 & 0 & 1.0311 & -0.9883 & -0.0074 & -0.1775 & -1.6428 & 1.6352 & -0.5880 & 0.1074 & -1.9514 \\ 0 & 0 & 0 & 0 & 1.0265 & 0.0255 & 0 & -1.7075 & -0.8249 & 0 & 1.1790 & -0.7365 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1.3197 & -1.2635 & 0 & 1.1235 & -0.4117 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.9107 & 0.0318 & -0.7197 & -0.8434 & 0.3578 & -1.8620 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.2534 & 0.5991 & 0 & 1.6019 & 0.8208 \end{bmatrix}$$

Table II.

$$T_4 = \begin{bmatrix} -0.7460 & -2.9175 & 0 & 0.7948 & -9.1495 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -2.9175 & 15.6712 & -1.9801 & -3.4101 & -5.8051 & 0.6073 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1.9801 & 0.4356 & 0 & -9.8485 & -1.8267 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.7948 & -3.4101 & 0 & 2.0894 & -10.8749 & 0 & -0.7082 & -2.9714 & 0 & 0 & 0 & 0 \\ -9.1495 & -5.8051 & -9.8485 & -10.8749 & -7.1446 & -4.2824 & -9.0859 & -4.5191 & -1.6397 & 0 & 0 & 0 \\ 0 & 0.6073 & -1.8267 & 0 & -4.2824 & 7.2264 & 0 & 2.6290 & 3.0429 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.7082 & -9.0859 & 0 & -2.9548 & -9.0392 & 0 & -1.4314 & -8.3386 & 0 \\ 0 & 0 & 0 & -2.9714 & -4.5191 & 2.6290 & -9.0392 & -1.2743 & 1.6406 & -10.2047 & -12.4107 & -1.6024 \\ 0 & 0 & 0 & 0 & -1.6397 & 3.0429 & 0 & 1.6406 & -1.5392 & 0 & -10.9757 & -3.5687 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1.4314 & -10.2047 & 0 & 2.9235 & -9.0150 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -8.3386 & -12.4107 & -10.9757 & -9.0150 & -4.5172 & -3.6301 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.6024 & -3.5687 & 0 & -3.6301 & 15.8208 \end{bmatrix}$$

Table III.

Since $|\Delta T|_\infty < \mu - 1$, $\alpha^1, ..., \alpha^4$ are still stable memories for system (2). A typical simulation run of equation (2) with DT given above is depicted in Fig. 8. In this figure, the noisy pattern is generated by adding to the desired pattern uniformly distributed noise defined on [-1,1]. Convergence occurred in 6 steps with h = 0.06.

Case IV: Symmetric design. With $\eta = 0.001$, starting with matrix $T_3 = T + \Delta T$ (where T was obtained from Case I and ΔT was obtained from Case III), we determined for this case that $v = 9.5512$, and we determined the symmetric matrix T_4, given in Table III, in four iterations.

A typical simulation run for system (1) with T_4 given in Table III is shown in Fig. 9. In this case, the noisy pattern was generated by adding Gaussian noise $N(0,1)$ to the desired pattern. Convergence occurred in 14 steps with $h = 0.06$.

Concluding Remarks

In the implementation of artificial feedback neural networks, *parameter perturbations, transmission delays*, and *interconnection constraints* are frequently encountered. These phenomena can potentially give rise to a degradation in the qualitative behavior of the neural networks.

In the present article we used as a vehicle of study the realization of associative memories by means of neural networks with Hopfield structure and with activation functions specified either by sigmoidal functions or by saturation non-linearities.

We studied the effects of parameter perturbations of neural networks by using the notion of robustness. For networks with sigmoidal functions we established necessary and sufficient conditions for robustness and we showed that when a neural network is robust and when the implementation parameter errors are reasonably small then for every desired stable memory of the ideal network, there will exist a corresponding actual stored stable memory for the implemented network. Furthermore, the errors in the memories can be estimated from the parameter errors. We also showed that networks with saturation non-linearities will always be robust and, as expected, networks with saturation functions will in general be less sensitive with respect to parameter perturbations than networks with sigmoidal non-linearities for activation functions. Furthermore, for neural networks with saturation non-linearities, we established an upper bound for the parameter perturbations which ensures the invariance of stable stored memories under perturbations.

In studying the effects of time delays on neural networks, we considered global and local results. For networks with a single delay and with multiple delays, we first established sufficient conditions for the global stability of the network which involve a bound for the time delays. We then showed that when these conditions for global stability are satisfied, the local qualitative properties of the stable memories of a network with delays and a corresponding network without delays are essentially identical.

We demonstrated how existing synthesis methods can be generalized to incorporate interconnection constraints, simultaneously taking into account parameter perturbations. This was accomplished by realizing associative memories in the form of bipolar vectors, utilizing Hopfield-like neural networks with saturation non-linearities as activation functions. We demonstrated the applicability of these results by means of specific examples. We did not address the effects of interconnection constraints on the memory storage capacity of a network. This important topic is beyond the scope of the present discussion.

Results of the type considered herein have clearly important practical implications. Furthermore, the methodology used in arriving at these results can readily be adapted to a variety of applications involving many different types of neural network structures.

References

[1] T.A. Burton, "Averaged Neural Networks," *Neural Networks*, vol. 6, pp. 677-680, 1993.

[2] G.A. Carpenter, M.A. Cohen, and S. Grossberg, "Computing with Neural Networks," Science, vol. 235, pp. 1226-1227, 1987.

[3] L.O. Chua and L. Yang, "Cellular Neural Networks: Theory," *IEEE Transactions on Circuits and Systems*, vol. 35, pp. 1257-1272, 1988.

[4] P.P. Civalleri, M. Gilli, and L. Pandolfi, "On Stability of Cellular Neural Networks with Delay," *IEEE Trans. Circuits Systems*, vol. 40, pp. 157-165, 1993.

[5] M.A. Cohen and S. Grossberg, "Absolute Stability of Global Pattern Formation and Parallel Memory Storage by Competitive Neural Networks," IEEE *Transactions on Systems, Man, and Cybernetics*, vol. 13, pp. 815-826, Sept./Oct. 1983.

[6] S.R. Das, "On the Synthesis of Nonlinear Continuous Neural Networks," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 21, pp. 413-418, March/April 1991.

[7] J.A. Farrell and A.N. Michel, "A Synthesis Procedure for Hopfield's Continuous-Time Associative Memory," *IEEE Transactions on Circuits and Systems*, vol. 37, pp. 877-884, July 1990.

[8] J. Hale, *Theory of Functional Differential Equations*, Springer-Verlag, 1977.

[9] U. an der Heiden, *Analysis of Neural Networks*, Springer-Verlag, New York, 1980.

[10] J.J. Hopfield, "Neural Networks and Physical Systems with Emergent Collective Computational Abilities," *Proc. Nat. Acad. Sci. USA*, vol. 79, pp. 2554-2558, Apr. 1982.

[11] J.J. Hopfield, "Neurons with Graded Response Have Collective Computational Properties Like Those of Two-State Neurons," *Proc. Vat. Acad. Sci. USA*, vol. 81, pp. 3088-3092, May 1984.

[12] B. Kosko, *Neural Networks and Fuzzy Systems*, Englewood Cliffs, NJ: Prentice-Hall, 1992.

[13] J.-H. Li, A.N. Michel, and W. Porod, "Qualitative Analysis and Synthesis of a Class of Neural Networks," *IEEE Trans. on Circuits and Systems*, vol. 35, pp. 976-986, Aug. 1988.

[14] J.-H. Li, A.N. Michel, and W. Porod, "Analysis and Synthesis of a Class of Neural Networks: Linear Systems Operating on a Closed Hypercube," *IEEE Trans. on Circuits and Systems*, vol. 36, pp. 1405-1422, Nov. 1989.

[15] D. Liu and A.N. Michel, "Sparsely Interconnected Neural Networks for Associative Memories with Applications to Cellular Neural Networks," *IEEE Trans. on Circuits and Systems-II: Analog and Digital Signal Processing*, vol. 41, pp. 295-309, Apr. 1994.

[16] D. Liu and A.N. Michel, *Dynamical Systems with Saturation Nonlinearities: Analysis and Design*, London: Springer-Verlag, 1994.

[17] C.M. Marcus and R.M. Westervelt, "Stability of Analog Neural Networks with Delay," *Physical Review A*, 39, pp. 347-359, 1989.

[18] C.M. Marcus and R.M. Westervelt, "Dynamics of Iterated-Map Neural Networks," *Physical Review A*, vol. 40, pp. 501-504, July 1989.

[19] C. Mead, *Analog VLSI and Neural Systems*, New York: Addison-Wesley Publishing Co., 1989.

[20] A.N. Michel and J.A. Farrell, "Associative Memories via Artificial Neural Networks," IEEE Control Systems Magazine, vol. 10, pp. 6-17, Apr. 1990.

[211 A.N. Michel, J.A. Farrell, and W. Porod, "Qualitative Analysis of Neural Networks," IEEE Trans. on Circuits and Systems, vol. 36, pp. 229-243, Feb. 1989.

[22] A.N. Niches and K. Wang, *Qualitative Theory of Dynamical Systems*, Marcel Dekker Inc.. New York, 1995.

[23] R.K. Miller and A.N. Michel, *Ordinary Differential Equations*, Academic Press, New York, 1982.

[24] R. Perfetti, "A Neural Network to Design Neural Networks," *IEEE Transactions on Circuits and Systems*, vol. 38, pp. 1099-1103, Sept. 1991.

[25] L. Personnaz, I. Guyon, and G. Dreyfus, "Information Storage and Retrieval in Spin-Glass Like Neural Networks," *J. Physique Lett.*, vol. 46, pp. L359-365, Apr. 1985.

[26] L. Personnaz, I. Guyon, and G. Dreyfus, "Collective Computational Properties of Neural Networks: New Learning Mechanisms," *Physical Review A*, vol. 34, pp. 4217-4228, Nov. 1986.

[27] F.M.X. Salam, Y. Wang and M.-R. Choi, "On the Analysis of Dynamic Feedback Neural Nets," *IEEE Transactions on Circuits and Systems*, vol. 38, pp. 196-201, Feb. 1991.

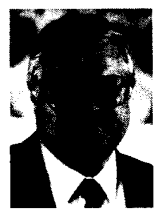
[28] M.J. Todd, *The Computation of Fixed Points and Applications*, Springer-Verlag, New York, 1976.

[29] K. Mang and A.N. Michel, "Robustness and Perturbation Analysis of a Class of Neural Networks," *Neural Networks*, vol. 7, pp. 251-260, 1994.

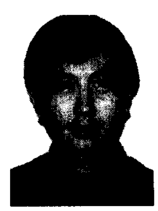
[30] H. Ye, A.N. Michel, and K. Wang, "Global Stability and Local Stability of Hopfield Neural Networks with Delays," *Physical Review E*, vol. 50, no. 5, pp. 4206-4213, Nov. 1994.

[31] H. Ye, A.N. Michel, and K. Wang, "Qualitative Analysis of Cohen-Grossberg Neural Networks with Multiple Delays," to appear in *Physical Review E*.

[32] G. Yen and A.N. Michel, "A Learning and Forgetting Algorithm in Associative Memories: The Eigenstructure Method," *IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing*, vol. 39, pp. 212-225, Apr. 1992.

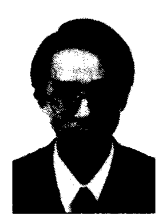


Anthony N. Michel (S'55-M'59-SM'79-F'82) received the Ph.D. degree in electrical engineering from Marquette University and the D.Sc. degree in applied mathematics from the Technical University of Graz, Austria. He has seven years of industrial experience. From 1968 to 1984, he was at Iowa State University, Ames, IA. From 1984 to 1988, he was Frank M. Friemann Professor and chairman of the Department of Electrical Engineering, University of Notre Dame, Notre Dame, IN. Currently, he is Frank M. Friemann Professor and Matthew H. McCloskey Dean of the College of Engineering at the University of Notre Dame. He is author and coauthor of three texts and several other publications. Michel is Vice President of Technical Affairs of the IEEE Control Systems Society.



Kainining Wang received the B.S. degree in statistics from the University of Science and Technology in China in 1982, the M.S. degree in applied mathematics from the Institute of Applied Mathematics, Chinese Academy of Sciences in 1985, and the Ph.D. degree from the University of Notre Dame in 1992. Currently he is a visiting assistant professor in the Department of Electrical and Computer Engineering at Wayne State University. He was a faculty fellow in the Department of Electrical Engineering at the University of Notre Dame from 1992 to 1994, and was a research fellow in the Institute of Economics, Chinese Academy of Social Sciences, from 1985 to 1986.

Derong Liu received the B.S. degree in mechanical engineering from East China Institute of Technology in 1982, the M.S. degree in electrical engineering from the Institute of Automation, Chinese Academy of Sciences in 1987, and the Ph.D. degree in electrical engineering from the University of Notre Dame in 1993. During his first year of graduate study at Notre Dame (1990-1991), he received the Michael J. Birck Fellowship.



Hui Ye was born in Hunan, the People's Republic of China, on May 11, 1968. He received the B.S. degree from the University of Science and Technology of China, and the M.S. degree from the University of Notre Dame, both in mathematics, in 1990 and 1992 respectively. He is currently working toward the Ph.D. degree in electrical engineering at the University of Notre Dame. His research interests include neural networks, control theory, non-linear systems with time delays, hybrid systems, and power systems.