Journal of Statistical Physics, Vol. 34, Nos. 3/4, 1984

Dynamics of Self-Organization in Complex Adaptive Networks

D. d'Humières[1,2] and B. A. Huberman[1]

Received August 20, 1983

We study the dynamical behavior of complex adaptive automata during unsupervised learning of periodic training sets. A new technique for their analysis is presented and applied to an adaptive network with distributed memory. We show that with general imput pattern sequences, the system can display behavior that ranges from convergence into simple fixed points and oscillations to chaotic wanderings. We also test the ability of the self-organized automaton to recognize spatial patterns, discriminate between them, and to elicit meaningful information out of noisy inputs. In this configuration we determine that the higher the ratio of excitation to inhibition, the broader the equivalence class into which patterns are put together.

KEY WORDS: Automata dynamics; self-organizing networks; learning and recognition.

1. INTRODUCTION

The problem of handling large amounts of redundant data and extracting relevant information from them lies at the heart of both pattern recognition automata and models of the higher brain functions, such as learning and associative memory. As such, they have been the focus of intense efforts aimed at designing algorithms and architectures. Among the many avenues being explored, a promising one resorts to local and parallel computation by arrays of processors with delocalized memories.(1-3)

In spite of all this work, little is known about the dynamics of complex networks and their behavior under general circumstances. Issues such as

[1] Xerox Palo Alto Research Center, Palo Alto, California 94304.

[2] Permanent address: Ecole Normale Supérieure, 24 rue Lhomond, 75231 Paris Cedex 05, France.

their self-organization, stability under parameter changes, and the extent of their parallel computing cannot be answered with ease, even in the simplest nontrivial cases. And yet, even partial answers to these questions are of relevance to VLSI design and the understanding of neural organization; the latter now emerging as one of the central problems of neurobiology.

Although part of the scarcity of answers is due to the lack of general enough experiments, a more serious problem has been posed by the absence of a methodology with which to systematically analyze experimental data. We believe that recent advances in dynamical systems theory can provide such a framework, and lead to a quantitative understanding of these very important issues. Moreover, complex adaptive networks with hierarchical structures lend themselves to techniques that have been developed in the study of nonlinear systems with few degrees of freedom.(4)

This paper reports on work that we have performed on the dynamical behavior of complex networks composed of interconnected cells arranged in a hierarchically layered structure. The systems that we dealt with are such that they possess a fixed wiring configuration but variable connection strengths, allowed to track the inputs so as to produce the best output according to given criteria. In particular, as the connectivity of the network changes when subjected to an input set, it is of interest to determine whether an asymptotic or "learning" behavior can take place. By this we mean a situation whereby the output produced by the system does not change much with further presentations of the same pattern sequence. Moreover, one would like to know how such convergence depends on parameters such as excitation, lateral inhibition, or connectivity.

In order to answer these questions, we first present a stroboscopic technique to analyze the time evolution of a layered adaptive network during *unsupervised* learning of periodic sequence of training sets (Section 2). The method is then applied to the experimental study of an adaptive network with distributed memory described in Section 3. We show that with general input pattern sequences, changing the relative levels of excitation versus inhibition leads to fixed points, oscillatory states, and chaotic wanderings during the adaptive process (Section 4). Moreover, using the self-organized network after the adaptive process has taken place, we study the size of equivalence classes as a function of excitation and inhibition. We also present preliminary results on pattern recognition of incomplete and noisy patterns (Section 5). A conclusion summarizes our findings and poses some questions for future research, and an Appendix provides the mathematical details of Section 3.

The results that we have obtained point to the usefulness of computer experiments in order to gain insights into issues relevant to self-organization in complex systems. Rather than optimizing an existing network or trying to closely model neurobiological systems, we have concentrated on the general dynamic properties of adaptive automata, a problem which underlies the behavior of many other machines.

2. STROBOSCOPIC DYNAMICS OF LAYERED NETWORKS

2.1. General Description of Adaptive Networks

Consider the general layered network shown in Fig. 1, having p layers, each layer containing n cells. Each layer l of the net is characterized at time t by vectors $\mathbf{I}_l(t)$ and $\mathbf{O}_l(t)$, made up of the inputs and outputs of the layer's cells, an array $S_l(t)$ which stores the information about the internal state of the network, and a transformation F relating the output of the layer

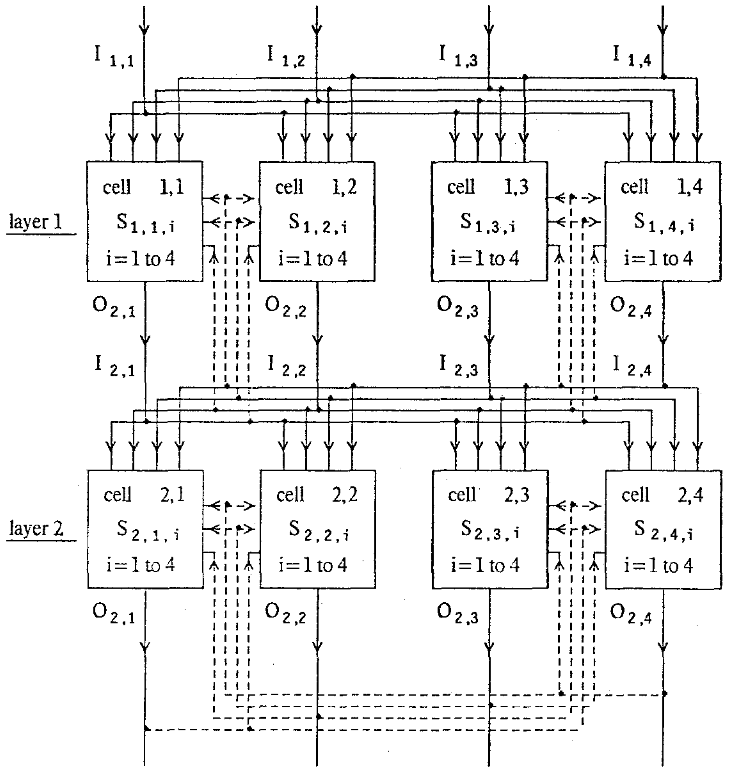

Fig. 1. A network with two layers and four cells per layer. The dashed lines show the feedback path during the adaptive phase.

at time $t + \tau$ to its input and its state at time t, i.e., $\mathbf{O}_l(t + \tau) = F(S_l(t), \mathbf{I}_l(t))$. Each cell i is a device with n afferent inputs stemming from the n outputs of the preceding layer, i.e., $\mathbf{I}_l(t) = \mathbf{O}_{l-1}(t)$, which in turn produces an output $\mathbf{O}_{l,i}(t)$ whose value depends on its internal state $S_{l,i}(t)$, the input values, and the propagation rules given by F. The inputs to the first layer are given externally, and the information is transmitted from layer to layer, down to the last layer, which we will call the output layer.

As long as the states of all the cells are fixed there are no fundamental differences between asynchronous or synchronous networks. This is no longer true however, if the states are allowed to change in time through an adaptive process. Since synchronous networks cover a wide range of applications and are easier to model and simulate than asynchronous ones, we will restrict our study to them. Moreover, we will assume that the external inputs of the first layer are stable for long enough to insure that both the state variables of the network and the output of the last layer are stable. We will also change the external inputs of the first layer at a fixed rate (define by an external clock with a period $T \gg \tau$). Under these conditions the network can then be sampled at the same clock rate in time intervals $k \cdot T$, which are multiples of T. Therefore transients can be ignored and $\mathbf{I}_l(t)$, $\mathbf{O}_l(t)$ and $S_l(t)$ can then be replaced by $\mathbf{I}_l^{(k)}$, $\mathbf{O}_l^{(k)}$ and $S_l^{(k)}$, respectively, with

$$\mathbf{O}_l^{(k)} = \mathbf{I}_{l+1}^{(k)} \tag{2.1}$$

$$\mathbf{O}_l^{(k)} = F\left(S_l^{(k)}, \mathbf{I}_l^{(k)}\right) \tag{2.2}$$

Any adaptive process consists of a feedback mechanism through which the different outputs can act back on the internal state of the network. Here we will restrict this process to feedback of the outputs of one layer on its own internal state, as shown by the dashed lines in Fig. 1. The adaptive process for the layer l is then determined by its initial state and by a transformation G_l, relating its state at sampled time $k + 1$, to its state and its output at sampled time k:

$$S_l^{(k+1)} = G_l\left(S_l^{(k)}, \mathbf{O}_l^{(k)}\right) \tag{2.3}$$

We should point out that the general network being considered here has a material connectivity (or wiring) independent of the adaptive process. The latter only changes the relative strength of the couplings as the input is changed, thereby producing an effective change in connectivity. This is to be contrasted with other automata where the rules are such that the wiring itself is allowed to change with the adaptive process.

2.2. Stroboscopic Dynamics

In principle, the time evolution of the network is completely given by the transformations F, G_l, and the input sequence. Unfortunately, with the exception of trivial transformations on networks with few elements, our knowledge about the dynamics of systems with many degrees of freedom does not provide many useful insights. However, the layered and hierarchical structure of the network allows us to consider as equivalent all states producing the same output at the last layer for a given input vector.[3] Thus, we only need to focus on the last output vector, thereby reducing the number of independent variables from $p \cdot n \cdot (1 + m \cdot n)$ to n (if the dimensionality of the state arrays S_l is $m \cdot n \cdot n$). For a network such as the one we studied ($n = 144$, $p = 3$, $m = 2$), this simplification reduces the number of independent variables (or degrees of freedom) by almost three orders of magnitude.

An additional simplification arises if the input patterns $\mathbf{I}_l^{(k)}$ are presented to the network in a periodic sequence, i.e.,

$$\mathbf{I}_l^{(k+K)} = \mathbf{I}_l^{(k)} \tag{2.4}$$

In this case we can use a method similar to the mappings at a period used in classical mechanics,[(4)] and which consists in sampling the state of the network at each period of the input sequence. This can be achieved by recording, for example, the output vector for a given input, or the output vector with maximum length at each period. Therefore, analyzing the periodically sampled data, it is easier to draw conclusions about the system's behavior. In particular, we have the following:

(i) One point in the sampled hyperspace will indicate a cycle, either a static fixed point, or a cycle with the same frequency as the input sequence.

(ii) Several points will indicate a more complicated motion, such as periodic behavior but with a fundamental period which is a harmonic of the input pattern sequence.

(iii) A closed trajectory will indicate quasiperiodic motion, i.e., dynamics in which the state of the network changes with a frequency which is incommensurate with that of the input pattern sequence.

(iv) A cloud of points will indicate the presence of chaotic behavior. By this, we mean dynamical behavior such that, although deterministic in origin, the system can be best described by probabilistic methods. As we show below, this behavior implies lack of convergence of the adaptive process although not necessarily poor flow properties of the input patterns.

[3] Thus we will consider as irrelevant any modification of the intermediate outputs or of the internal states which do not produce any change on the last output.

In spite of the simplification brought about by this periodic sampling of the network, we are still faced with the problem of following a vector in time in a high dimension hyperspace (144 in our experiment). As we are interested in the existence of fixed points for the system, one possible solution is to measure some distance between the successive outputs and these fixed points. Since unfortunately we have no *a priori* idea of the fixed points distribution in the hyperspace for a given set of parameters, we can instead measure the distance between two successive outputs produced by the same input pattern. With this technique, the absence of convergence for this quantity will signal the lack of any fixed points for the system. On the other hand, its convergence to zero will be a strong indication of the existence of fixed points.

In what follows, the distance between two vectors $\mathbf{V}$ and $\mathbf{W}$ (either inputs or outputs), will be defined as one minus the usual coherence function between the two vectors, i.e., their inner product divided by their Euclidian measures:

$$\langle \mathbf{V}, \mathbf{W} \rangle = 1 - \mathbf{V} \cdot \mathbf{W} / (\|\mathbf{V}\| \cdot \|\mathbf{W}\|) \tag{2.5}$$

Since $\mathbf{V} \cdot \mathbf{W} = \|\mathbf{V}\| \cdot \|\mathbf{W}\|$ if and only if $\mathbf{V} = c \cdot \mathbf{W}$ (c and $\|\mathbf{W}\|$ nonzero), two vectors differing by only a multiplicative constant will be considered as equal.[4]

Thus, to describe the network we will measure for each period K of the training sequence, the maximum and the minimum value of the quantities:

$$\langle \mathbf{O}_p^{(k+\kappa \cdot K)}, \mathbf{O}_p^{(k+(\kappa+1) \cdot K)} \rangle \tag{2.6}$$

evaluated at the last layer for $k = 0$ to $K - 1$ as a function of time κ, with the time unit equal to the period of the input sequence.

3. EXPERIMENTAL ADAPTIVE MACHINE

In what follows we will describe a particular adaptive network which we used in order to test the theoretical ideas presented in Section 2. Among the very many cell structures,[5-10] we chose the one shown in Fig. 2, which was introduced by Fukushima.[7] The details of the mathematical formulation of this algorithm are given in the Appendix. Here we will focus on its basic structure, along with the numerical values used for the simulations.

The cells we used were made of two spatial filters simultaneously fed by the n inputs to the layer. Each filter consists of n multipliers followed by a summing unit and is controlled by an adaptive module into which the

[4] One can easily check that this distance is not a true measure as it does not satisfy the triangular inequality.

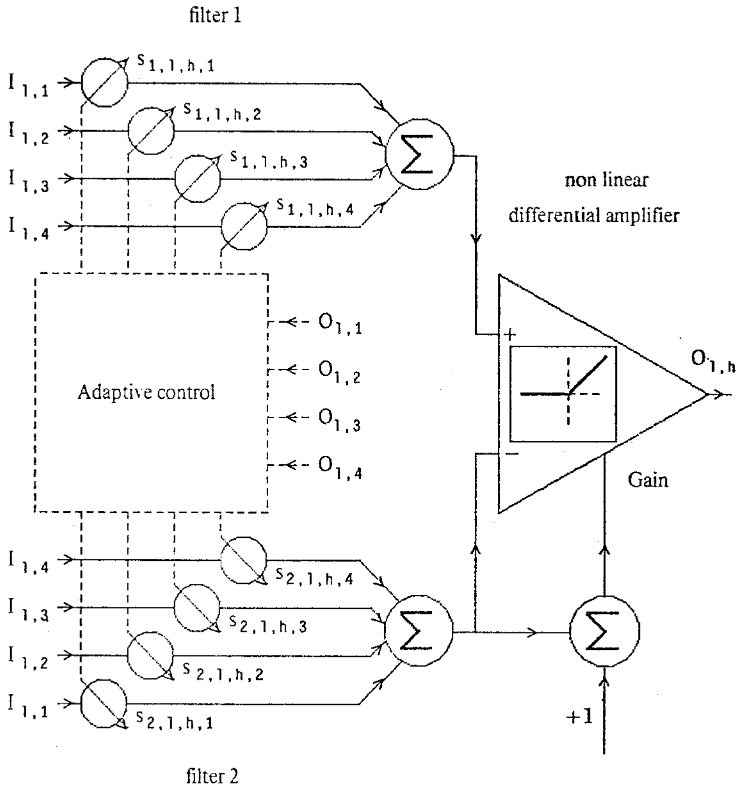

Fig. 2. Schematic representation of a cell of the network.

outputs of all the cells are fed back, as shown by the dashed lines of Fig. 2. The outputs from the two filters are then compared by a nonlinear differential amplifier acting as a rectifier (a threshold device with a fixed threshold equal to zero). In addition, the output of the second filter added to 1 sets the inverse of the amplifier gain.

The effect of these filters is determined by the actual adaptive process. In a pattern recognition automaton, the goal is to increase the distance between the outputs produced by the different training patterns and to broaden the equivalence classes associated with them. To achieve this, the adaptive process is implemented as follows. The output of each cell is locally compared to outputs from other *given* cells in the *same layer*. The states of the cells producing local maxima are then changed by adding a part of the input vector to the coefficients of the first filter, thus producing filters better matched to this input. In this way the first set of filters acts as a template comparator and is referred to as the exictatory set of connections. At the same time, a constant term related to the average input value is added to the coefficients of the second filter, leading to a measure of the "linear power" of all positive inputs, along with a normalization of the output vectors. This is done in order to make them independent of the number of times the first filter has been modified by the same input. This second set of coefficients will be referred to as the inhibitory set of connections. Finally, the gain of the differential amplifier is set by a

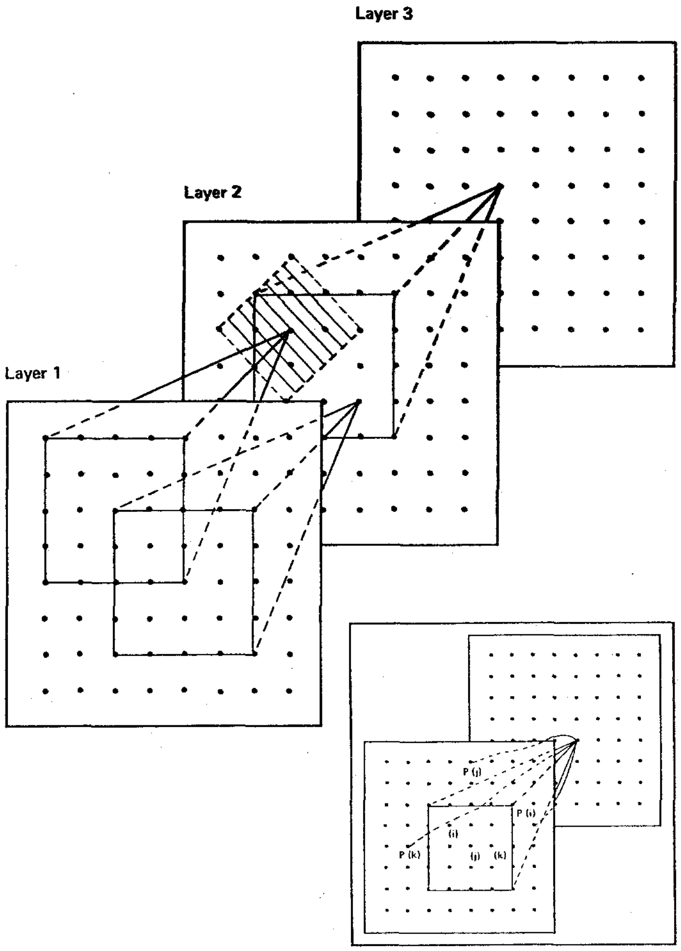

Fig. 3. Actual implementation of the network. All the cells inside the squares act as inputs for the cell at the top of the cones. The insert shows the random set of connections for the cells as indexed by a random permutation. The shaded diamond shows the range of comparison between outputs of the same layer during the adaptive process.

quantity Q_0 which measures the ratio of excitation to inhibition, or equivalently, the amount of information flowing through the network, as shown in the Appendix.

The automaton was then simulated in the following configuration:

(1) A three-dimensional structure made up of three layers ($p = 3$) with the input and output vectors folded into 12 by 12 square matrices (i.e., $n = 144$).

(2) The connections between the inputs of one layer and the outputs of the preceding one were separated into two sets. Through a first set of connections, each cell of a given layer was connected to its 25 nearest neighbors in the previous one (see Fig. 3). A second set of connections had a random character and it connected each cell to another 25 cells of the previous layer through assignments made with a random number generator. This random set of connections, wired into the network and therefore independent of the adaptive process, was first introduced by Fukushima(7) in order to extend the range of interactions between cells in different layers. These cell–cell interactions were implemented through *given* matrices B_l, which thus have only 50 nonzero coefficients.

(3) The range of comparison between outputs of a layer during the adaptive process was determined by the outputs of the 12 nearest neighbors (a 5×5 diamond configuration within the same layer, as shown in Fig. 3 by the shaded area).

(4) Each layer was split in two parts, the first one remaining as an adaptive layer, and updated by the output of a second layer with the same structure but *given* filters. This second step provided an edge enhancement process of the intermediate output vector, as shown in the Appendix. As far as lateral inhibition was concerned, each cell was connected to the 49 nearest neighbors of the adaptive layer in a similar fashion as the adaptive connections.

4. RESULTS OF ADAPTIVE EXPERIMENTS

With the specifications given above, we studied the dynamics of the network for two training sets of input patterns as a function of the ratio of excitation to inhibition Q_0. The first training set consisted of all the horizontal and vertical full lines (12 dots long) which could be arranged into the square input matrix, whereas the second set was composed of the 26 capital letters A to Z plus the ten digits 0 to 9, arranged in 9×11 matrix within the 12×12 array.

Typically, the maximum and the minimum of the quantities defined by Eq. (2.6) were measured over many periods (from 60 to 600) of the input pattern sequence and their decimal logarithm was plotted as a function of time (in units of pattern sequences). This is shown in Figs. 4–6. The lower curves correspond to the minimum distance measured for some patterns and the upper one denotes the maximum distance.

For both sets of patterns, the best convergence properties for the network, as measured by these curves, were found for $Q_0 \simeq 2$. As expected, the time to reach a fixed point was longer for the more complicated set of input patterns. As Q_0 was decreased or increased away from that value, we found out that the convergence of the adaptive process was altered, as shown in Figs. 4–6.

In particular, with the input set composed of lines, we discovered (Fig. 5d) a periodic behavior in a very narrow range of parameter values, i.e., $1.48 < Q_0 < 1.51$. It was characterized by rapid oscillations of the distance

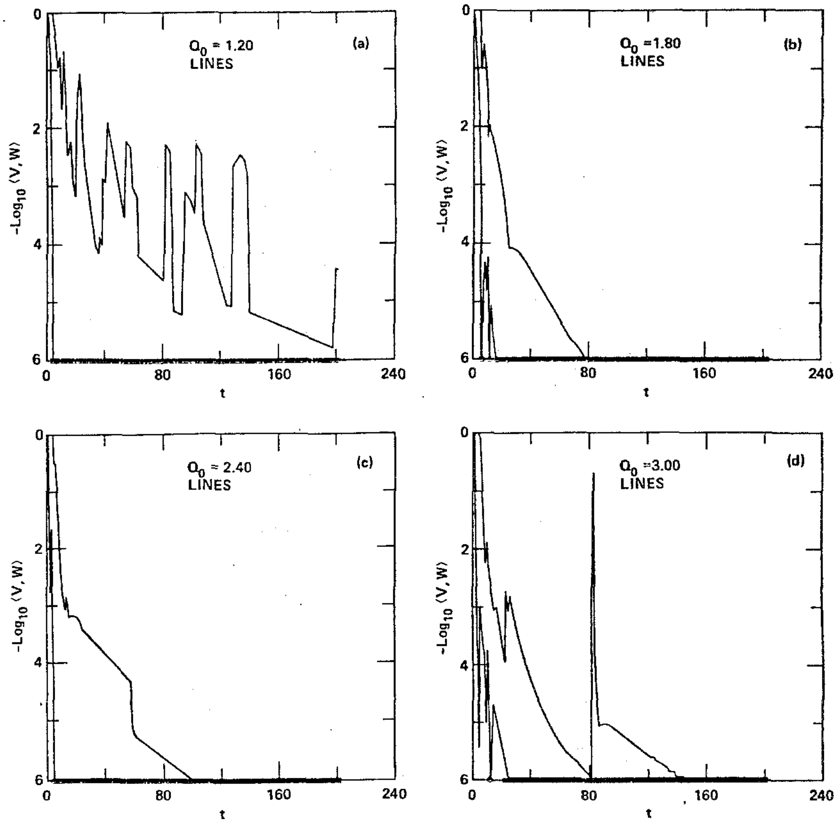

Fig. 4. The distance $\langle V, W\rangle$ as a function of time for a training set of lines. Data obtained for values of $Q_0 = 1.2$, 1.8, 2.4, and 3.0. The time unit for this figure, and the following two, is defined as the time to process a complete set of input patterns.

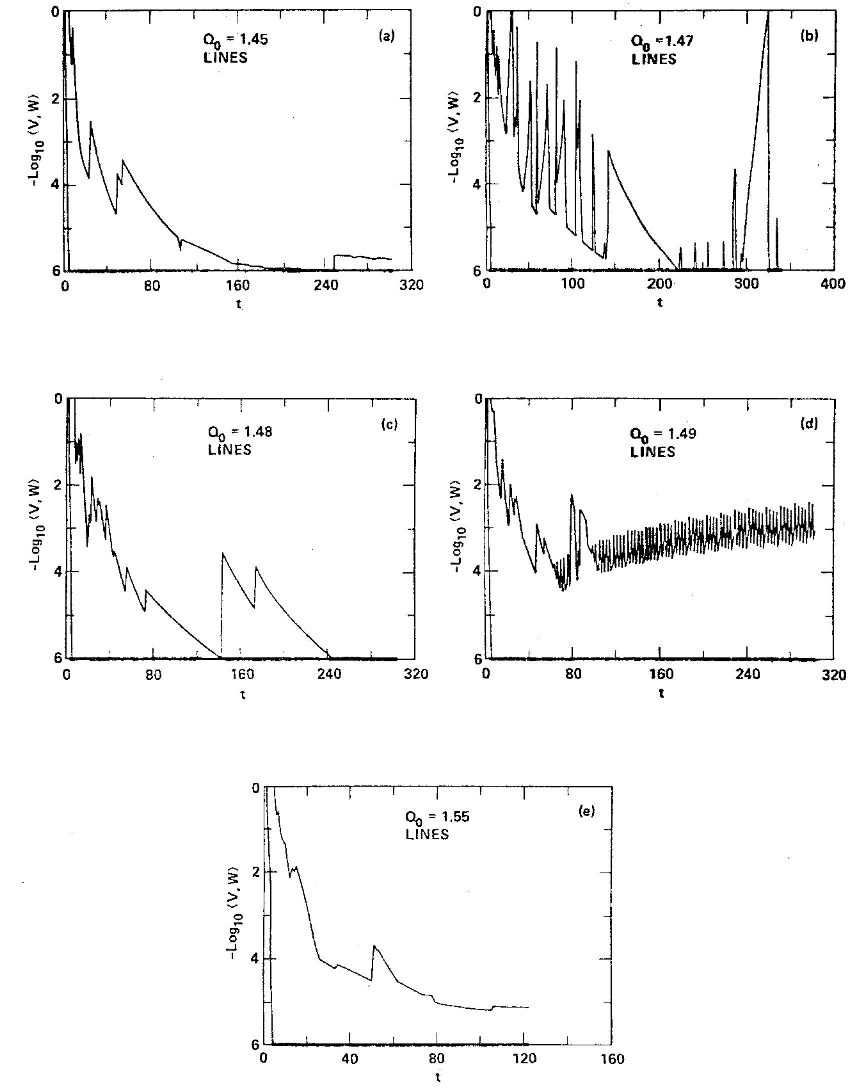

Fig. 5. The distance $\langle \mathbf{V}, \mathbf{W} \rangle$ as a function of time for a training set of lines. Data obtained for $Q_0 = 1.45$, 1.47, 1.48, 1.49, and 1.55.

after many passes of the complete set of inputs. This process represented a dynamical state of the network such that the relative strengths of its connections changed in a cyclic manner as the training set was presented over and over again. Furthermore, for values of Q_0 between 1.46 and 1.48, we found (Fig. 5b) a chaotic regime, with the distance varying erratically

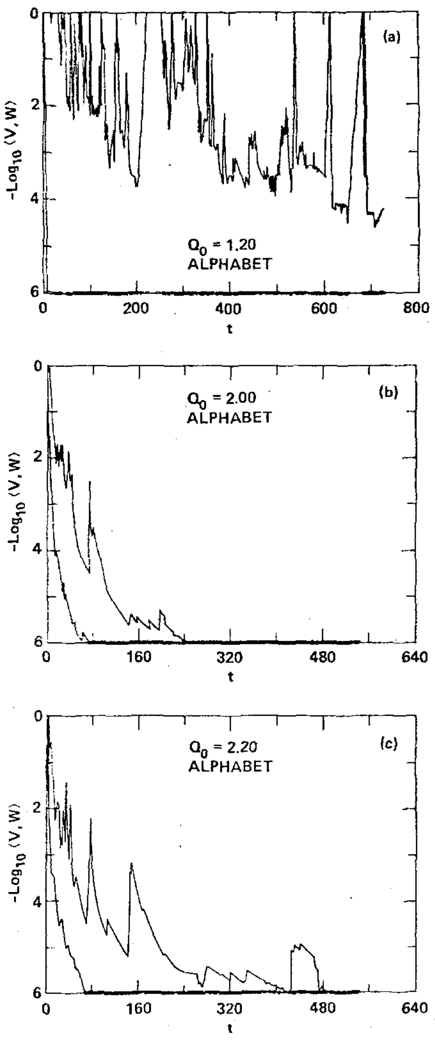

Fig. 6. The distance $\langle \mathbf{V}, \mathbf{W} \rangle$ as a function of time for the alphabet training set. Data obtained for $Q_0 = 1.2$, 2.0, and 2.2.

between zero and small values for several periods of the input sequence. We should also point out that these periodic and chaotic behaviors were entangled with regimes for which the network flowed toward a fixed point, the ranges of existence for each of them being very narrow.

Another interesting phenomenon is illustrated in Fig. 4d. For long times the network shows monotonic convergence towards a self-organized state with a simple fixed point, only to start unraveling itself at later times. Although all these phenomena exist for different input sequences, the exact numerical values of Q_0 associated with them depend on the actual input sequence; see, for example, the adaptive behavior for an alphabetic training set depicted in Fig. 6a.

These results are to be contrasted with dynamical systems with few degrees of freedom, in which the sequences of attractors one observes are both simpler and immune to external perturbations.[11] The reason for this difference seems to be due to the presence of a few patterns which during the adaptive process start producing weaker outputs. As this process continues, a particular pattern ends up producing a zero output vector, thus leading to a bootstrapping procedure to recover from this situation. This in turn produces a change in the distributed memory of the network in such a way so as to take it away from its fixed point. One can conclude from this observation that minor perturbations can eventually drive a complex network away from its fixed points.

5. EXPERIMENTS ON THE SELF-ORGANIZED NETWORK

In what follows we will describe experiments which are performed on the network *after* the adaptive process took place. These experiments used the network in a pattern recognition mode as a probe of its final state, thus studying the filtering properties of the network and how they were related to the training set of patterns.

With the state of the network encoded in its final state arrays S_l, we computed, and stored as templates, all the output patterns $\mathbf{O}_p^{(k)}$ produced by the different input patterns $\mathbf{I}_l^{(k)}$ of the training set. Typical output patterns are shown in Fig. 7; as can be seen, they range from having only one nonzero component (Figs. 7a and 7b) to having several cells with positive values (Figs. 7c and 7d). We should point out that most of the output patterns obtained for all values of Q_0 showed this latter behavior.

For each couple of training input patterns $\{\mathbf{I}_l^{(k)}, \mathbf{I}_l^{(k')}\}$ we then measured both their mutual distance, as defined by Eq. (2.5), i.e.,

$$d_i = \langle \mathbf{I}_l^{(k)}, \mathbf{I}_l^{(k')} \rangle \tag{5.1}$$

and the respective mutual distance of their output vectors at the last layer:

$$d_o = \langle \mathbf{O}_p^{(k)}, \mathbf{O}_p^{(k')} \rangle \tag{5.2}$$

The output correlations, as a function of Q_0, produced by the pattern S with other letters of the alphabet, are summarized in Table I. As Q_0 increases from 1.2 to 2.2 one can easily see that the network evolves from a state with very sharp discrimination between similar patterns, to a state

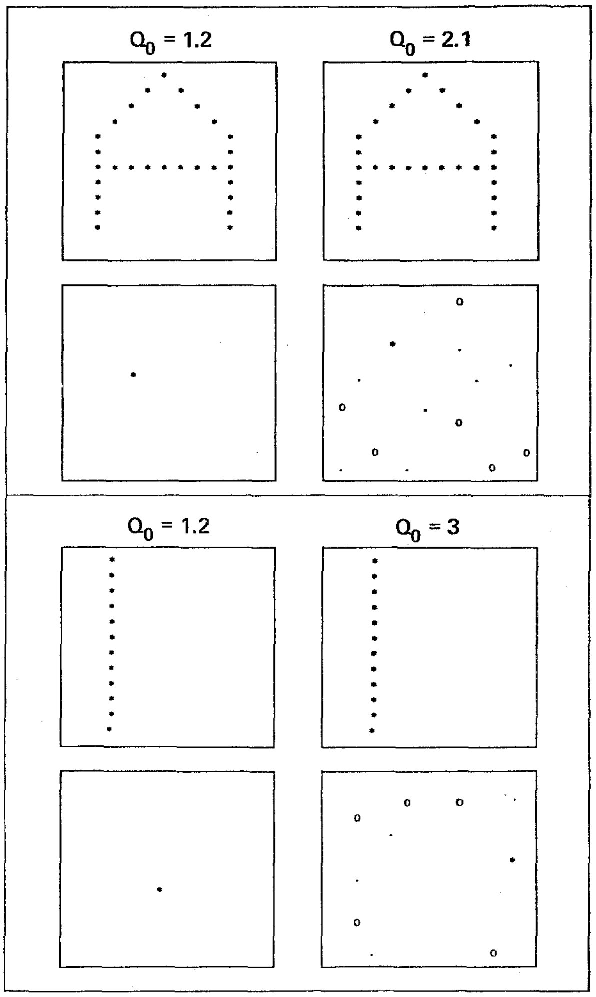

Fig. 7. Input and output patterns for a character and a line. The stars ($*$) represent the position of the strongest output, the circles ($\circ$) the positions of outputs greater than average, and the dots ($\cdot$) the positions of outputs less than average.

with broad class aggregation (or equivalence classes), as shown by the underlined values.[5]

Moreover, using this procedure for input patterns other than the training set, we found out that this result also holds for incomplete patterns

[5] The equivalence classes are arbitrarily defined in such a way that the distance between the last pattern in a class and the first one excluded is a maximum.

Table I. The input and output distances between the pattern S and other letters of the alphabet for different values of Q_0. The distances equal to 1 (orthogonal patterns) are omitted for clarity. The numbers with a star correspond to distances between patterns lower at the output than at the input (worse discrimination after processing). The underlined values denote patterns belonging to the same equivalence class.

		Output					
					$Q_0 =$		
Pattern	Input	1.2	1.4	1.6	1.8	2.0	2.2
8	0.067	0.700	0.560	$\underline{0.065}$*	$\underline{0.037}$*	$\underline{0.002}$*	$\underline{0.001}$*
9	0.101	0.559	0.623	$\underline{0.106}$	$\underline{0.092}$*	$\underline{0.012}$*	$\underline{0.010}$*
6	0.101	0.672	0.344	$\underline{0.095}$*	$\underline{0.051}$*	$\underline{0.140}$	$\underline{0.016}$*
B	0.230			0.952	0.828	$\underline{0.140}$*	$\underline{0.080}$*
C	0.238		0.786	0.760	$\underline{0.103}$*	$\underline{0.267}$	$\underline{0.081}$*
3	0.245			0.811	0.718	$\underline{0.130}$*	$\underline{0.080}$*
G	0.262		0.781	0.747	$\underline{0.039}$*	$\underline{0.278}$	$\underline{0.105}$*
O	0.273		0.814	0.792	$\underline{0.153}$*	$\underline{0.296}$	$\underline{0.108}$*
5	0.274				0.885	0.771	0.386
Q	0.320	0.800	0.987	0.856	$\underline{0.282}$*	$\underline{0.338}$	$\underline{0.183}$*
D	0.388			0.970	0.794	$\underline{0.222}$*	$\underline{0.068}$*
2	0.396	0.701		0.808	$\underline{0.275}$*	$\underline{0.201}$*	$\underline{0.198}$*
E	0.397				0.935	0.812	0.480
P	0.444					0.936	0.830
R	0.456					0.993	0.809
F	0.500						0.991
U	0.500			0.967	0.819	0.829	0.800
...	...	...	...	...	...	...	...
Z	0.593						0.544
...	...	...	...	...	...	...	...
W	0.678	0.827				0.998	0.967
7	0.647						0.760

or noisy inputs as well. Quite generally, we concluded that *the higher the ratio between excitatory and inhibitory connections, the larger the equivalence class* and therefore the easier for a given input to produce an output correlated with the learnt patterns.

A possible explanation of these results, along with those of the previous section, can be derived from the fact that Q_0 measures the differential gain of the cell and thus the amount of information flowing through the network. For low values of Q_0, only few elements of the total input information are propagated from layer to layer. This leads to a high network selectivity and a consequent destruction of important elements of the input patterns. On the other hand, for large values of Q_0 most of the information is propagated through the network, leading to a buildup of filters matched to some average input. This in turn results in broad aggregation of the patterns as processed by the machine. These results, which are also relevant to cognitive processes in neurobiology, emphasize the importance of quantitative measurements in the study of pattern recognition by automata.

6. CONCLUSION

Complex automata are structures situated in between the dynamics of few degrees of freedom and the simplifying disorder encountered in many-body systems like gases. As such they pose a special challenge when trying to understand their dynamical properties as a function of given parameters and inputs.

In this paper we have shown that it is indeed possible to study in a quantitative fashion the dynamics of their self-organization. Through the introduction of a general methodology, we were able to obtain crisp information on issues that are central to the understanding of data processing by machines and brains. Also, by performing experiments on a particular nonlinear adaptive network, we uncovered a rich variety of behaviors and quantified them as a function of excitation, inhibition, and connectivity. In that fashion we discovered that in addition to regimes where asymptotic learning can take place, there exist scenarios characterized by periodic oscillations and chaos. Moreover, experiments on the recognition properties of the automaton led to an understanding of the dependence of equivalence classes on excitation and inhibition. Generally speaking, we concluded that the higher the ratio of excitation to inhibition, the broader the equivalence class into which patterns are lumped together. This finding might be of relevance to both pattern recognition machines and neurobiology.

Last but not least, we should mention the issue of universality, i.e., to what extent our results depend on both the particular set of training patterns or the automaton being simulated. Whereas they indicate that the behavior encountered in this study does not depend on a particular pattern sequence or type, we have only tentative conclusions concerning independence of network architecture. Although we believe that our findings are likely to be found in any layered automaton obeying local computational rules, more experiments will be necessary to test this hypothesis.

ACKNOWLEDGMENTS

We have benefited from useful conversations with T. Hogg and M. Kerszberg. D. d'Humières would like to thank the Xerox Palo Alto

Research Center for its hospitality during his stay. This work was partially supported by O.N.R. contract N00014-82-0699.

APPENDIX

In this Appendix we present the mathematical details of the algorithm used to implement the cell structure shown in Fig. 2. Since each cell needs two filters and n coefficients per filter, the state array S_l of layer l can be separated into two $n \times n$ submatrices $s_{1,l}$ and $s_{2,l}$, whose lines $s_{1,l,i}$ and $s_{2,l,i}$ store the filter coefficients of the cell i. The transformation F is then given by

$$F(S, \mathbf{V}) = \Phi(s_1 \times \mathbf{V}, s_2 \times \mathbf{V}) \tag{A1}$$

where Φ is a transformation of $\mathbf{R}^n \times \mathbf{R}^n$ into $\mathbf{R}^n$ (where $\mathbf{R}$ is the real field) such that the components U_i of $\mathbf{U}$, V_i of $\mathbf{V}$, and W_i of $\mathbf{W} = \Phi(\mathbf{U}, \mathbf{V})$ are related by

$$W_i = \max\{0, (U_i - V_i)/(1 + V_i)\} \tag{A2}$$

and where $\times$ denotes the usual matrix product. The adaptive process for the layer l is then given by an initial state defined by both setting all coefficients $S_l^{(0)}$ to zero, and by

$$S_l^{(k+1)} = \left\{s_{1,l}^{(k+1)}, s_{2,l}^{(k+1)}\right\} = G_l\left(S_l^{(k)}, \mathbf{I}_l^{(k)}\right) \tag{A3}$$

with

$$s_{1,l}^{(k+1)} = s_{1,l}^{(k)} + \Psi_l\left(\mathbf{I}_l^{(k)}, \mathbf{O}_l^{(k)}\right) \tag{A3a}$$

and

$$s_{2,l}^{(k+1)} = s_{2,l}^{(k)} + \Xi_l\left(\mathbf{I}_l^{(k)}, \mathbf{O}_l^{(k)}\right) \tag{A3b}$$

where Ψ_l and Ξ_l are two transformations of $\mathbf{R}^n \times \mathbf{R}^n$ into $\mathbf{R}^{n \cdot n}$ such that the elements $M_{i,j}$ of $M = \Psi_l(\mathbf{U}, \mathbf{V})$, and $N_{i,j}$ of $N = \Xi_l(\mathbf{U}, \mathbf{V})$, are related to two *given* parameters q_1 and q_2 ($q_1 > q_2$), to the components U_i of $\mathbf{U}$, V_i of $\mathbf{V}$ and to the elements $B_{l,i,j}$ of *given* $n \times n$ matrices B_l, through the relations

$$M_{i,j} = q_1 \cdot \delta_i \cdot B_{l,i,j} \cdot U_j \tag{A4}$$

$$N_{i,j} = q_2 \cdot (\mathbf{M}_i \times \mathbf{U})/(\mathbf{B}_{l,i} \times \mathbf{U}) \tag{A5}$$

with

$$\delta_i = 1, \qquad \text{if} \quad V_i = \max\{V_j \,;\, j \in E_{l,i}\} \tag{A6}$$

and

$$\delta_i = 0, \qquad \text{if} \quad V_i < \max\{V_j \,;\, j \in E_{l,i}\} \tag{A7}$$

where the matrices B_l provide local intercell connections between consecutive layers when needed, and where the terms $E_{l,i}$ are *given* sets of integers between 1 and n which determine the outputs from other cells in the *same layer* l to be locally compared to the output of cell i.

If the inputs remain always positive, the coefficients of the state array grow indefinitely in time as the training sequence is repeated. At the same time, the modifications induced by each pattern become decreasingly important. Within this context, q_1 determines the relative amount of change. Simultaneously, V_i becomes large compared to 1 in Eq. (A2), and $Q_0 = 1/q_2$ then provides a measure of the differential amplifier gain, or of the amount of information flowing through the network.

Lastly, we should mention that the exact implementation of the algorithm is slightly more complicated than that described above. This stems out of the need to deal with the problems of bootstrapping the network out of a state characterized by a null output for any input. We thus replaced Eqs. (A4) and (A5) by

$$M_{i,j} = q \cdot \delta_i \cdot B_{l,i,j} \cdot U_j \tag{A8}$$

$$N_{i,j} = q \cdot \delta_i \cdot (B_{l,i} \times \mathbf{U}) \tag{A9}$$

whenever V_i was zero and still a maximum. In this case the cells in the neighborhood of a given one produce a zero output for nonzero input, and it becomes necessary to build up the connections with equal weights for the two filters. This particular process is continued with the same input pattern until the network produces a nonzero output.

Within this scheme, Eq. (A1) is also replaced by

$$F(S,\mathbf{V}) = \Phi(\Phi(s_1 \times \mathbf{V}, s_2 \times \mathbf{V}) \cdot f \times \Phi(s_1 \times \mathbf{V}, s_2 \times \mathbf{V})) \tag{A1$'$}$$

where f is a *given* $n \times n$ matrix, usually labeled lateral inhibition in brain modeling. This is equivalent to splitting each layer in two parts, the first one remaining as an adaptive layer updated by the output of a second one having the same structure but *given* filters. The first filter selects the input with the same index as the cell (identity transfer matrix), and the coefficients of the second filter are *given* by f. This second step provides an edge enhancement process of the intermediate output vector. For example, if f is a bidiagonal matrix filled with $1/2$, then $\mathbf{V} - f \times \mathbf{V}$ is the discrete first-order approximation to the derivative of $\mathbf{V}$ along its components ($\mathbf{V}$ is essentially the sampling of a continuous function). This same process can construct higher order derivatives or many similar output transformations.

REFERENCES

1. J. A. Feldman and D. H. Ballard, *Cognitive Science* **6**:205 (1982) and references therein.
2. T. Kohonen, *Associative Memory* (Springer, New York, 1978); see also *Parallel Models of*

Associative Memory, E. E. Hinton and J. A. Anderson, eds. (L. Elbaum Associates, Hillsdale, New Jersey, 1981).

3. G. Edelman and B. Mountcastle, *The Mindful Brain* (MIT Press, Boston, 1978); W. J. Freeman, in *Synergetics of the Brain*, H. Haken, ed. (Springer, New York, 1983).
4. V. I. Arnold, *Mathematical Methods of Classical Mechanics* (Springer, New York, 1978).
5. F. Rosenblatt, *Principles of Neurodynamics* (Spartan Books, Washington, D.C., 1962); M. Minsky and S. Papert, *Perceptrons* (MIT Press, Cambridge, 1969).
6. J. S. Koford and G. F. Groner, *IEEE Trans. Information Theory* **12**:42 (1966).
7. K. Fukushima, *Systems Computer Controls* **6**:15 (1975); *Tech. Monograph (N.H.K)* **30**:178 (1981).
8. M. Osborne, *IEEE Trans. Comp.* **C-26**:1302 (1977).
9. I. Morishita and H. Yajima, *Kybernetik* **11**:154 (1972).
10. R. Takiyama, *Pattern Recognition* **15**:405 (1982).
11. J. P. Crutchfield, D. Farmer, and B. A. Huberman, *Phys. Rep.* **92**:45 (1982).