ENGINEERING APPLICATIONS OF SPIN GLASS CONCEPTS

I. GUYON, L. PERSONNAZ, P. SIARRY, G. DREYFUS
Ecole Supérieure de Physique et de Chimie Industrielles
de la Ville de Paris
10, rue Vauquelin
75005 Paris
FRANCE

I - INTRODUCTION :

The very large research effort - both theoretical and experimental - which has been devoted to the understanding of spin glasses during the past years has had two largely unexpected consequences in the field of engineering : first, the emergence of the technique of simulated annealing, which has already had a strong impact on combinatorial optimization and its applications, and, secondly, a new wave of interest in neural networks, which turn out to be promising new tools for performing high-level functions in information processing.

Several papers of the present book deal with basic aspects of the application of statistical physics concepts to complex optimization problems ; similarly, new developments in the theory of neural networks are presented, both in connection with biological modelling and with more general aspects of information storage and retrieval. The purpose of this paper is to present some applications of these two fields. The first part of the paper will be devoted to an application of simulated annealing to the Computer-Aided Design of electronic circuits ; other applications of this technique will also be mentioned ; in the second part, we shall present two new learning rules for networks of formal neurons, which are directly aimed at applications such as the automatic recognition of handwritten characters.

II - ENGINEERING APPLICATIONS OF SIMULATED ANNEALING :

1) Simulated annealing and the Computer-Aided Design of electronic circuits :

The technique of simulated annealing has emerged from the very difficult problems encountered in trying to optimize the design of electronic circuits[1,2] ; this is still the field in which it is in most current use. In this section, we shall describe the problems encountered in automating the design of hybrid circuit modules, and the performances achieved by using simulated annealing. The optimization problem to be solved is the placement of the components, viewed as rectangular blocks of various sizes, given the "net-list", i.e., the list of the connections which are to be made between the various blocks present on the circuit. The subsequent step is the automatic drawing of the connections themselves. The introduction of a "temperature" in this context is very natural, as illustrated on Figure 1 : given a "random" initial placement and wiring, which is obviously not optimum, an ideal Computer Aided Design program should be able to turn it into a regular placement, with straight connections, thereby performing a "transition" between a disordered state and an ordered state.

Hybrid circuit technology is an intermediate technology between the familiar printed circuit technique and the sophisticated and expensive integrated circuit technology. A module consists in an insulating ceramic substrate, on which components such as capacitors, diodes, transistors and integrated circuit chips are bonded, either in miniature packages, or naked (for integrated circuits) ; resistors and connections are made of resistive inks and are silk-screened onto the surface. The typical length in hybrid technology is 50 µm, as compared to 1 mm for printed circuits and 1 µm for integrated circuits.

In a typical electronic circuit design process, the first step is the schematic capture, which results in a graphics file containing the symbolic description of the circuit, and in several text files including the net-list which is necessary for the placement and routing phases. For the placement itself, the geometrical characteristics of the blocks are extracted from a data base containing the components used in the design of the modules. The placement obeys two kinds of rules :

- the *design rules* are intended to guarantee the proper operation of the circuits : examples of such rules are minimum component spacing, minimum spacing between components and connections, etc. These rules *must* be satisfied at the end of the

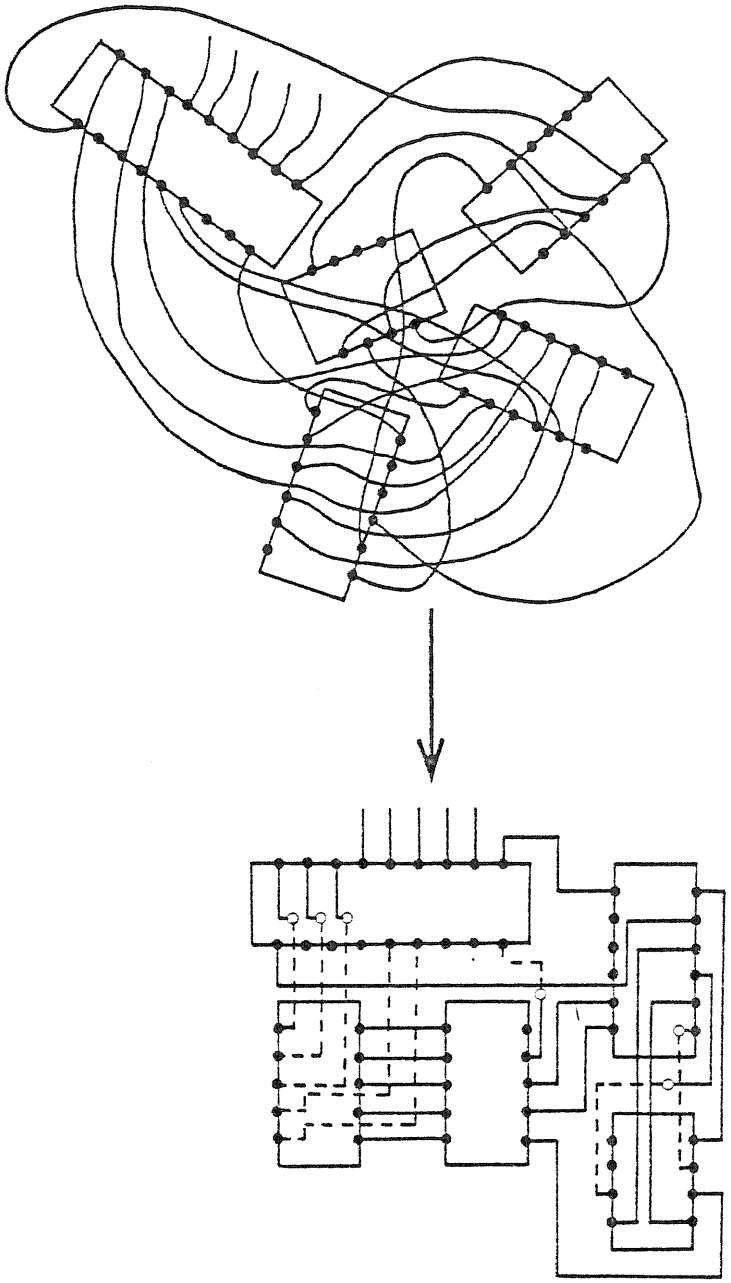


FIGURE 1 :
The design of an electronic circuit viewed as a disorder to order transformation.

placement and routing phase

- the *know-how rules* have been designed empirically by engineers in order to facilitate the subsequent routing phase. These rules may be violated locally at the end of the placement phase.

The main difference between the placement of hybrid circuits and the floor-planning of integrated circuits is the following : in integrated circuit technology, the cells which are to be placed cannot overlap, whereas hybrid circuit technology does allow overlaps : a resistor, being made of a resistive paste, is completely flat, so that a packaged integrated circuit, for instance, may be placed above it ; this is true only for resistors that do not need any dynamic adjustment by laser trimming.

Therefore, the placement program has to take into account both the design rules and the know-how rules, with the above-mentioned increased complexity for block overlaps. Moreover, since placement by simulated annealing is a time-consuming process, the placement obtained should allow a very easy routing, with only minor changes to the placement, whereas, in standard placement and routing processes, the routing phase may alter the placement extensively.

The necessary ingredients for a simulated annealing placement program are :

- an initial placement,
- an initial temperature,
- elementary moves defining a topology in state space,
- a cost function,
- an annealing schedule.

Two possibilities exist for the initial conditions : one may either use a random initial placement and a high initial temperature, or choose a realistic initial placement and use the simulated annealing technique, starting at a relatively low temperature, for improving this initial situation. The second alternative was chosen, the initial placement being made by a constructive procedure, whereby the most highly connected components are placed first, then the second most connected chips, and so on. This algorithm is very fast and gives an initial placement which, in general, satisfies the design rules and violates the know-how rules extensively.

The cost function included several terms, the leading terms being the connection length, the overlaps between blocks and the design rules. Other terms took the know-how rules into account, the respective weights being such that the algorithm first took care of the design rules, and subsequently of the know-how rules, just as a design

engineer would do.

The annealing schedule used basically the standard temperature variation, with a geometrical decrease by a factor of the order of .9 ; however, this schedule is adaptive since it takes into account the depth of the minima encountered during the evolution in state space.

As an illustration of the results which were obtained, Figures 2 and 3 show the initial and final states, respectively, for a simple module. The initial state was designed manually ; the total value of the cost function was of the order of 16,000 , including a contribution of 9,000 from the connection length and a contribution of 7,000 from the violations of the know-how rules. The cost function was decreased by a factor of 2, mostly through a drastic decrease in the violations of the know-how rules, the contribution of which dropped to a value of 10, but also through a decrease of the connection length by approximately 15%. It is interesting to note that the automatic procedure complies with the know-how rules better than the experts themselves ! The required CPU time in this example was 3 minutes of Amdahl V7.

Figure 4 shows a more complicated case in which the ability to perform the routing operation is obviously critical. In general, the choice of the cost function allowed the drawing of the connections with only minor changes to the placement. In all cases, the routing was done by conventional maze-running algorithms.

In conclusion, simulated annealing has turned out to be an efficient tool for the design of hybrid circuits : the problem with this technology is not a matter of number of components, as is the case for integrated circuits, but a matter of complex design rules. The algorithm has been adapted successfully to the special features of this technology.

2) Other engineering applications of simulated annealing :

Engineering applications of simulated annealing can be divided broadly into two classes :

- optimization problems,
- inverse problems.

The essential difference between the two classes of problems is the following : in optimization problems, one searches an acceptable optimum among a large number of possible optima corresponding to similar values of the cost function ; all optima having the same energy are considered equivalent. Conversely, in inverse problems, there is only one acceptable solution, namely, the real cause of the observed phenomena ; in

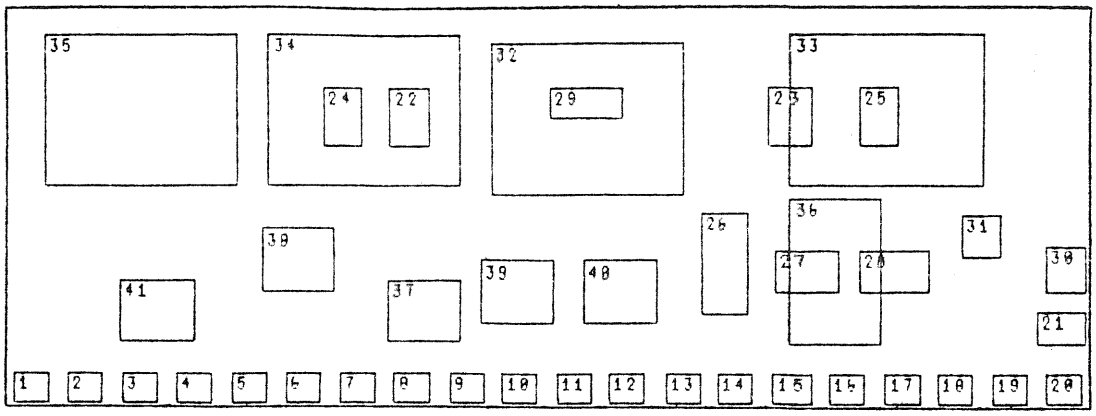


FIGURE 2 :
Initial configuration of a hybrid module (designed manually).
Total cost function : 16,000.
Connection length : 9,000.
Overlaps : 7,000.

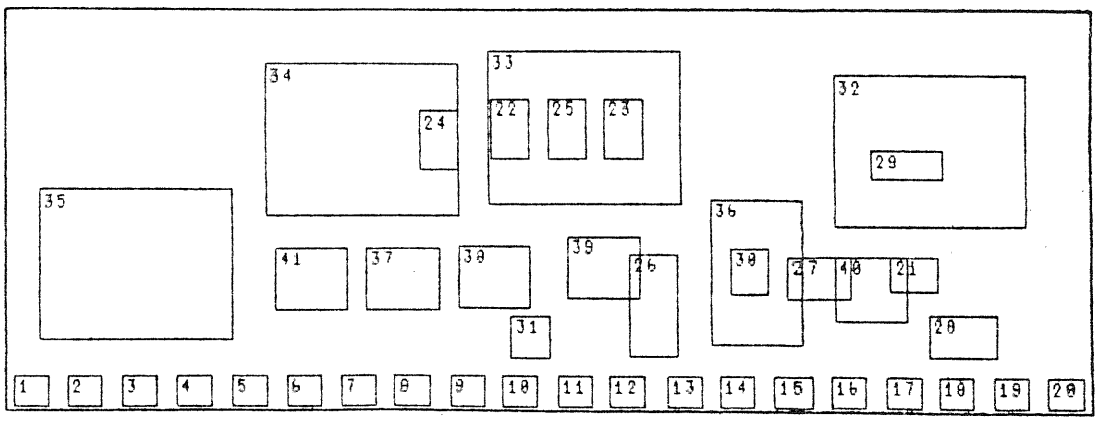


FIGURE 3 :
Final configuration of the above hybrid module (after annealing).
Total cost function : 8,000
Overlaps : 10.

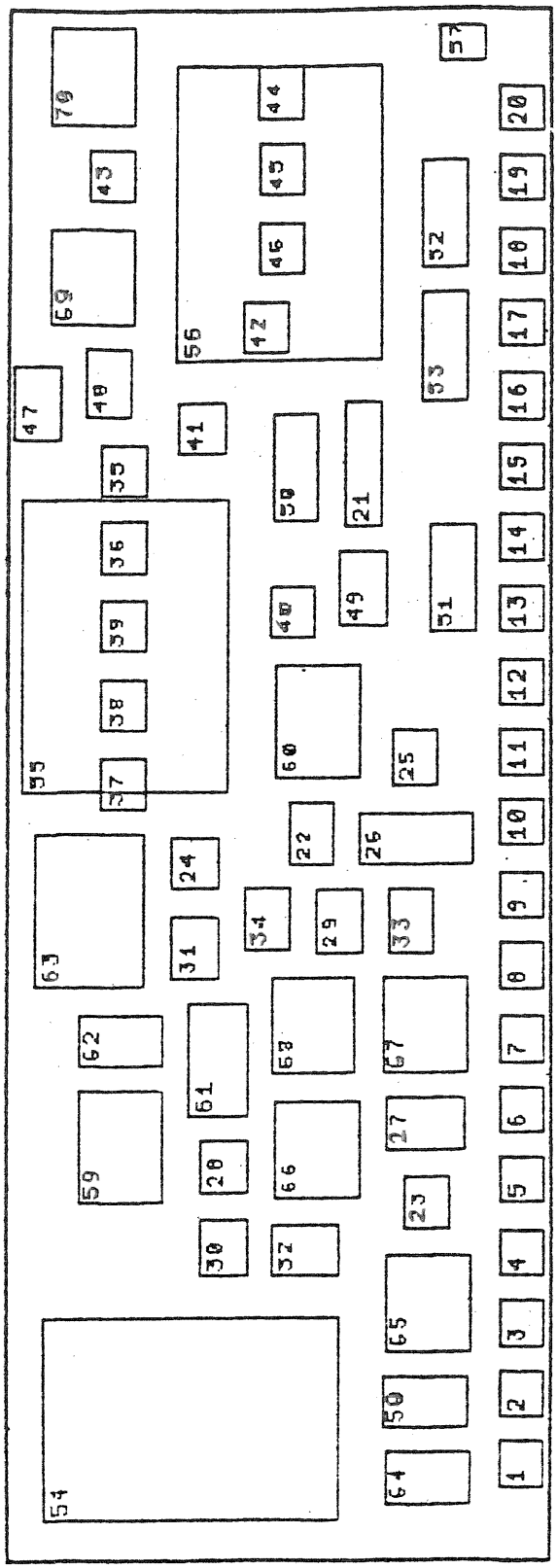


FIGURE 4 :
A "difficult" case for routing.

image restoration, for instance, one tries to find the original image, not any image giving the same cost function. Therefore, inverse problems have specific features which standard optimization problems do not exhibit.

a/ Inverse problems :

The most spectacular application of simulated annealing to inverse problems is probably the interpretation of seismic data by D. Rothman[3]. The problem is to reconstruct the geological structure of the terrain from information gathered from the propagation of sound waves in the structure. In the same spirit, applications in tomography by Paxman et al.[4] use simulated annealing in order to reconstruct 3D objects from 2D images. This is done by making an initial guess of the actual object, alter it randomly, compute the resulting image and minimize the difference between the computed and real 2D images.

Another inverse problem is the deblurring of binary images[5]. A general framework was proposed by Geman et al.[6], using a Bayesian approach for estimating the noise and the image.

b/ Optimization problems (other than placement and routing in electronics) :

Simulated annealing has been used in a variety of fields ; one can make the general statement that whenever an optimization problem can be solved by iterative improvement, it can be solved efficiently by simulated annealing. Other problem-specific strategies may be more efficient, but the strength of simulated annealing lies in the simplicity of its implementation and the variety of problems that it can solve.

Several problems in operations research have been successfully attacked by this technique : cloth cutting[7], task scheduling[8], the planning of office buildings[9], the warehouse problem (using the grand canonical ensemble)[10], etc...

Finally, two applications in electronics can be mentioned : one of them aims at simplifying the construction of electronic circuits in wire-wrapping technique [11] the other consists in optimizing the symbolic drawing of electronic schematics in order to improve their legibility[12].

3) Conclusion :

The technique of simulated annealing is gaining wide acceptance because of its

success both on "model" problems such as the traveling salesman problem, and on practical applications ; it is one of the manifestations of the fact that there is a strong interrelation between the concepts of statistical physics and those of combinatorial optimization. From a practical standpoint, developments may be expected in two directions : first, there is a clear need for problem-independent results on the convergence of the method, the annealing schedule[13], the influence of the cost function, etc. ; secondly, the parallelization of the algorithm may be necessary for very complex problems which require a large computing power[14-16]. Anyway, simulated annealing is already present in the toolbox of combinatorial optimization and operations research.

III - NEURAL NETWORKS AND INFORMATION PROCESSING :

The renewed interest in the collective computational properties of assemblies of simple neuron-like elements, triggered by J. Hopfield's work[17], has spurred developments in two directions : biological modelling and information processing ; obviously, the distinction is not quite as clear-cut as that, and many investigations are aimed at bridging the gaps between these fields. One important distinction between the two approaches is the "local" or "non-local" nature of the learning rules ; this issue will be discussed below, and in a more detailed fashion in Ref. 18 .
In the following, we show that neural networks can be used efficiently to perform pattern recognition, or, more generally, as distributed associative memories. Such systems are highly parallel and provide collective decisions distributed both in space (each neuron takes a decision), and in time (the state of the network evolves until it reaches a fixed point, which characterizes the final decision). These networks exhibit learning abilities since their structural parameters can be computed, from predetermined items of information, so as to impose dynamic behaviour constraints in state space.

We consider, a fully connected network of n deterministic Mc Culloch-Pitts formal neurons operating in parallel with period τ , without "sensory inputs". The coupling coefficients will be called C_{ij} and the thresholds will be taken equal to zero. The state

of a neuron i at time t , $\sigma_i(t)$, is a binary variable with value +1 or -1 , which depends on the state of the other neurons at time $t-\tau$ in the following way :

$$\sigma_i(t) = \mathrm{sgn}(v_i(t-\tau)) \qquad \text{if } v_i(t-\tau) \neq 0$$
$$\sigma_i(t) = \sigma_i(t-\tau) \qquad \text{if } v_i(t-\tau) = 0 \qquad (1)$$

where $v_i(t-\tau) = \sum_{j=1}^{n} C_{ij}\,\sigma_j(t-\tau)$ is the membrane potential of neuron i at time $t-\tau$.

Matrix notations can be used because of the parallel iteration dynamics : if $\sigma(t)$ is the vector whose components are the $\sigma_i(t)$ and $C = \{C_{ij}\}$, the network computes its (n-dimensional) "vector potential"

$$\mathbf{v}(t-\tau) = C\,\sigma(t-\tau) \ ,$$

and finds its next state vector $\sigma(t)$ after the threshold operation (1).

In the following, we address the problem of computing the matrix C so as to impose given dynamical constraints in state space.

1) " Digging holes" in state space :

In this section we show that it is possible to store any set of prototype states as fixed points of the dynamics (states invariant in time)[18]. Moreover the following desirable features are obtained :

- the prototype states act as attractors ;
- the non-prototype fixed points are useable ;
- no cycle occur.

Assume that we want to impose p prototype states σ^1, σ^2... σ^p as *fixed points* , which will be shown later to be attractors. The coupling coefficients must satisfy the following system of n.p inequalities :

$$\Big(\sum_j C_{ij}\,\sigma_j^k\Big)\,\sigma_i^k \geq 0 \qquad i=1, \ldots n \ ; \ k=1, \ldots p$$

but it is sufficient to solve the following linear system of n.p equations :

$$\sum_j C_{ij}\, \sigma_j^k = \sigma_i^k \qquad i=1, \ldots n \ ; \ k=1, \ldots p$$

The above condition can be written in matrix form :

$$C\,\Sigma = \Sigma$$

where $\Sigma = [\sigma^1, \sigma^2 \ldots \sigma^p]$ is the (n,p) matrix whose columns are the prototype vectors.

This equation has an infinite number of solutions[20] :

$$C = \Sigma\,\Sigma^I + B\,(I - \Sigma\,\Sigma^I)$$

where Σ^I is the pseudoinverse of Σ , B is an (n,n) arbitrary matrix, and I is the identity matrix.

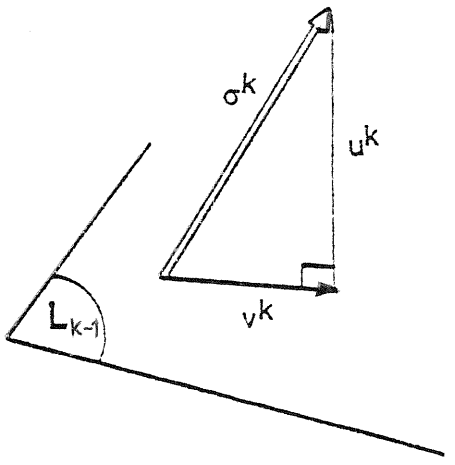
As far as associative memory properties are concerned, the most appropriate solution is obtained for B = 0 :

$$C = \Sigma\,\Sigma^I \qquad (2)$$

It is the orthogonal projection matrix into the subspace spanned by the p prototype vectors. However, other choices for matrix B have proved to be useful for other purposes[21]. Relation (2) will be referred to, in the following, as the *projection rule*.

The coupling matrix C can be computed iteratively by introducing sequentially the prototype vectors, as usual in most learning processes ; suppose that k-1 patterns have been learnt and that we want to learn a new pattern, the increment of matrix C will be:

$$\Delta C(k) = \frac{\mathbf{u}^k\,\mathbf{u}^{kT}}{\|\,\mathbf{u}^k\,\|^2}$$



where $\mathbf{u}^k = \sigma^k - \mathbf{v}^k$, with $\mathbf{v}^k = C(k-1)\,\sigma^k$.

L_{k-1} is the subspace spanned by the k-1 previously learned patterns.

An important issue in the choice of learning rules is their local nature. If biological plausibility is of importance, the learning rule must be local : the computation of a

coupling coefficient C_{ij} should require only informations available locally at neurons i and j. If, conversely, one is interested in devices, the local nature is usually irrelevant ; indeed, the *projection rule* is not local in general. However, for weakly correlated prototype vectors, it can be approximated by the following local rule :

$$\Delta C_{ij}(k) = (1/n).(\sigma_i^k \sigma_j^k - \sigma_i^k v_j^k - v_i^k \sigma_j^k) \quad .$$

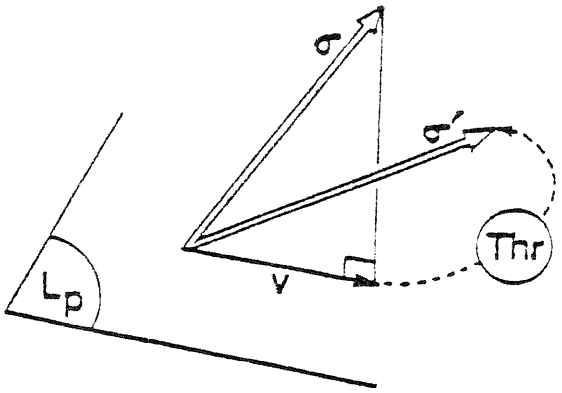
The first term is just the classical "Hebb's rule", inspired by the work of the neurophysiologist D. O. Hebb[22] ; the last two terms take into account the information that has been learnt previously.

The study of the dynamics shows that a network designed after the *projection rule* (relation (2)) exhibits the desired associative properties mentioned at the beginning of this section. First, we define the following Lyapunov function (which is similar to the energy of spin systems) :

$$E(\sigma) = -(1/2) \sum_{i,j} C_{ij} \sigma_i \sigma_j = -1/2 \, (\sigma^T C \, \sigma) \quad .$$

This is an ever decreasing function during the free evolution of the system[18]. Therefore, any evolution of the network will end up in a fixed point, so that *no cycle can occur* . Moreover, since the prototype states have the lowest possible energy, they act as *attractors* of the system. Thus, the state space is partitioned into several basins of attraction, the bottom of which are the fixed points.

The following Figure shows a geometrical interpretation in Euclidean space of one iteration.



Starting from a state σ, the network performs two minimizations in order to reach its next state σ' :

i) $\mathbf{v} = C \, \sigma$ is the orthogonal projection of σ into the subspace L_p spanned by the p

prototype vectors. Therefore, it is the linear combination of prototype states which is closest to the subspace L_p ; in other words, $\mathbf{v}$ is the best possible approximation of the unknown pattern in terms of a linear combination of the learnt patterns.

ii) σ' is the vector belonging to $\{-1,+1\}^n$ which is closest to $\mathbf{v}$.

Consequently, if σ itself is the vector of $\{-1,+1\}^n$ which is closest to $\mathbf{v}$, σ is a fixed point ; *the fixed points are "thresholded" linear combinations of the prototype states* . We shall see in the following that these states play a crucial role in the decision taking process. The dynamics of the present model of neural networks has been studied extensively (albeit in a slightly modified form) by H. Sompolinsky et al[19].

The *projection rule* can be used for error correction purposes, as illustrated in reference 18. An example of a classification task will be shown in the section devoted to character recognition.

2) "Digging valleys" in state space :

In this section we address the more general problem of imposing a set of transitions between states. The learning rule which is proposed allows to memorize either stable states, as the *projection rule* does, or sequences of transitions and/or cycles. It provides a new tool to perform associations.

Assume that we want to impose p one-step transitions in state space defined by :

$$\sigma^k \rightarrow \sigma'^k \qquad k=1, \ldots p \quad .$$

The coupling coefficients must satisfy a system of n.p inequalities :

$$(\sum_j C_{ij}\, \sigma_j^k)\, \sigma_i'^k > 0 \qquad i=1, \ldots n \ ; \ k=1, \ldots p \qquad (3)$$

and, following a similar derivation as in the previous section, we compute matrix C by the following relation :

$$C = \Sigma' \Sigma^I \qquad (4)$$

where $\Sigma' = [\sigma'^1, \sigma'^2 \ldots \sigma'^p]$,

which is a solution of (3) if $\Sigma' \Sigma^I \Sigma = \Sigma'$. Relation (4) will be called the *associating rule* ; it minimizes the euclidean norm of the error matrix $C\,\Sigma - \Sigma'$. Notice that if all the

imposed transitions are such that $\sigma'^k = \sigma^k$, the *associating rule* reduces to the *projection rule* . Similarly, the *associating rule* can be put into an iterative form.

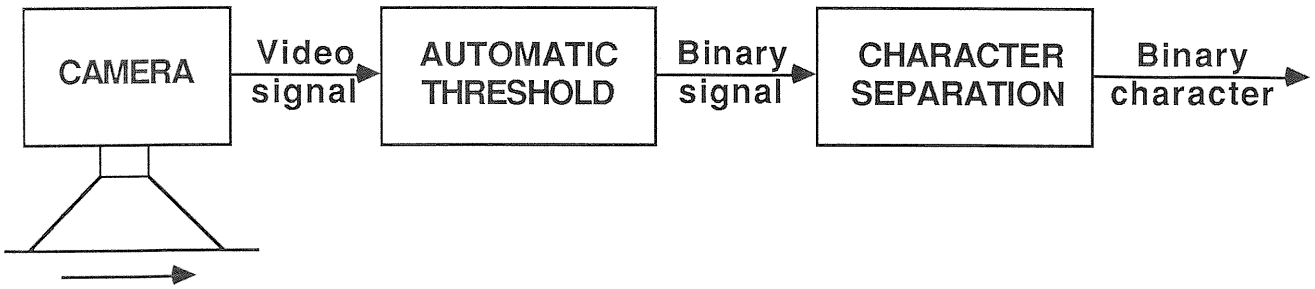
In reference 18, as well as in the section devoted to character recognition, we have used the *associating rule* for classification tasks. Several informations are associated to their class by a one-step transition and the class itself is made a fixed point. After learning, the network is able to classify successfully many unknown informations, but, if the information is too distorted, the network goes to a non-prototype fixed point σ^* or "garbage state". However, if this nonprototype fixed point is undesired, it can be eliminated by imposing one extra transition :

$$\sigma^* \rightarrow \sigma^k$$

In this sense the *associating rule* can "dig valleys" in state space.

3) Character recognition :

Handwritten character recognition is an engineering problem which has been extensively studied for two decades. Software implementations of handwritten character recognition systems require a high computing power. We show in the following, that neural networks can be used as simple parallel computing devices which perform character recognition. We have tested the ability of neural nets, designed after the above two learning rules, to solve an actual handwritten numeral recognition problem. More specifically, we are interested in the recognition of postal codes.



a/ Description of the data acquisition system.

A solid state camera gives a video signal which is the image of a postal code. This signal is preprocessed by an automatic threshold device, the purpose of which is to compensate for the contrast variations. Thus, a binary image of the postal code is obtained. Each numeral is subsequently isolated by a hardwired device and is

subsequently stored in a 32×32 bit memory (RAM). Each numeral is automatically positioned in the upper left corner of the memory. A numeral is thus represented by a vector of 1024 binary components (pixels).

Several classical methods in character recognition involve various other preprocessing tasks which aim at suppressing the undesired sensitivity to geometrical transformations (translations, rotations, etc.). A well-known technique is based on the extraction of binary features describing the topology of the drawing (bays, loops, intersection of line segments, etc.), which gives a second representation tolerant to distortions, translations, slight rotations, style and scale variations of the characters. We show in the following that a proper choice of the prototype patterns allows recognition without any preprocessing other than the automatic thresholding : the 1024 binary pixel representation of characters can be used directly. This straightforward representation is probably non-optimal ; nevertheless, it helps testing the efficiency of the proposed learning rules, and gives satisfactory results as far as character recognition is concerned.

b/ Use of a coding field.

Automatic character recognition is essentially a classification problem, in which we are interested only in identifying unknown characters. Thus, we are led to specialize some neurons for an identification task, which means that we add a *coding field* to the information vector[23].

The proposed classifier is based on the following construction of the state vectors. They have two distinct fields : the first field, denoted by $\mathbf{e}$, is devoted to the pattern itself (1014 components), and the second, denoted by $\mathbf{c}$, to the class code (10 components). We consider several prototypes per class, corresponding to several styles for a given numeral. During the learning phase, each prototype pattern $\mathbf{e}^k$ belonging to the class r ($1 \leq r \leq 10$) is associated to a coding field $\mathbf{c}^r$. The code has -1's only, except for the r^{th} position ; for example, the prototypes of the 4^{th} class are coded by the field:

$$\mathbf{c}^4 = [-1 \ -1 \ -1 \ +1 \ -1 \ -1 \ -1 \ -1 \ -1 \ -1]^T$$

During the classification phase, an unknown pattern is presented to the network with a code field having -1's only. After some parallel iterations, a fixed point is reached. The unknown pattern is attributed to the class corresponding to the code of the fixed point.

The decision process can be analyzed as follows : at each iteration, the network first computes the potential vector which is a linear combination of the prototypes :

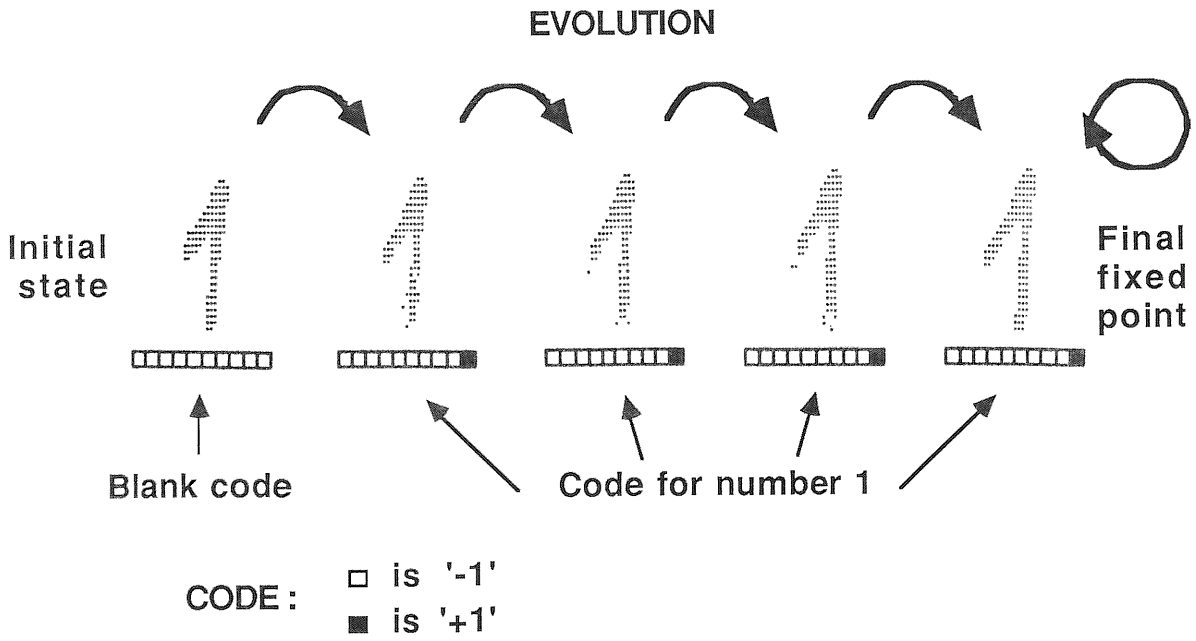
$$v = \sum_{k=1}^{p} a_k \sigma^k,$$

the value of a_k representing the weight of the prototype σ^k in the linear combination of the prototype states (which is the best approximation of the unknown pattern at this step of the free evolution of the network). The choice of the coding scheme leads to the following decision process after thresholding : in the coding field, the state of the neuron related to the class r takes the value +1 if :

$$\sum_{k \in \text{class } r} a_k \; > \; \sum_{k \notin \text{class } r} a_k .$$

The coding field of the final fixed point can have :

- only one "+1" component : an unambiguous association is achieved;
- no "+1" component, or several "+1" components : the network gives an ambiguous response to an ambiguous piece of data, so that the character is not recognized.



A point of fundamental importance in this scheme is the following : nonprototype attractor states which are mixtures of various styles of one character have the code of that character. This is exemplified below.

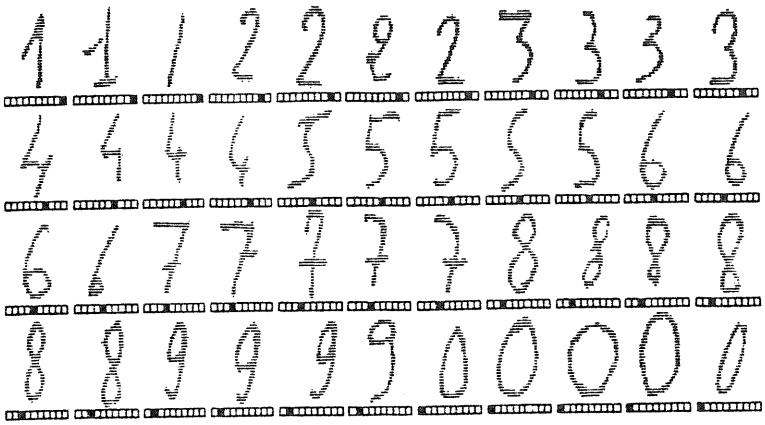
c/ Results.

The classifiers have been simulated on the CIRCE computers (9080 NAS and 3090 IBM).

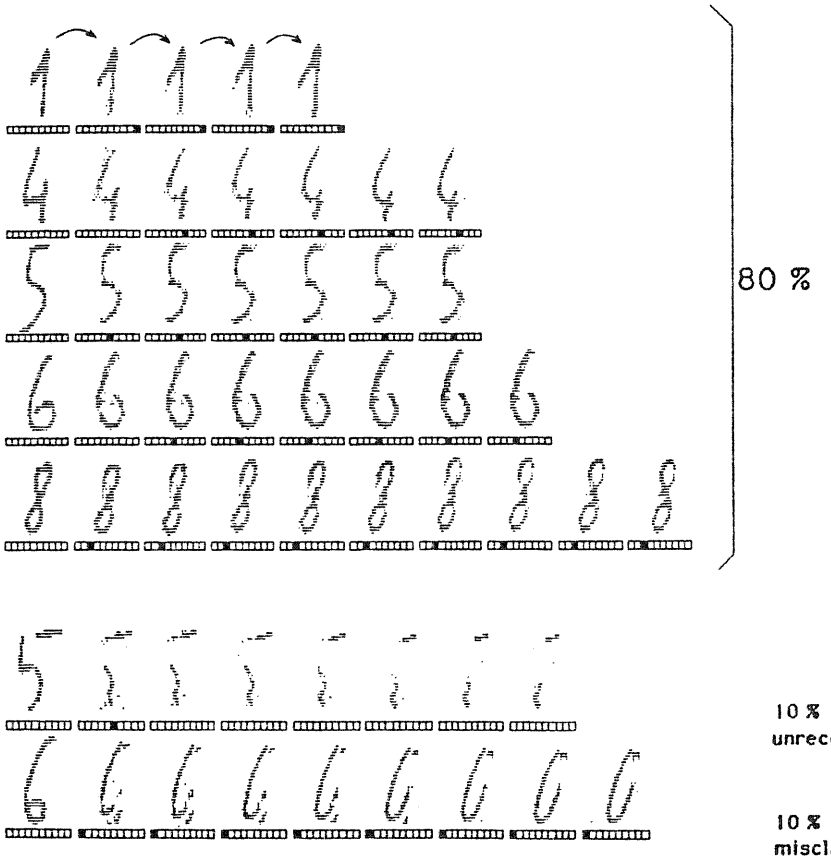
Projection rule.

After the memorization, with the projection rule, of 44 prototype handwritten numerals with their 10-bit code (figure 5.a), 250 unknown patterns were presented to the network; approximately 80% of them were well recognized (with the right code), 10% misclassified (with a wrong code) and 10% unrecognized (with a meaningless code). Some examples of evolutions are given in figure 5.b. One should notice that some of the correctly recognized characters lead to fixed points which are not present in the training set, but look like the patterns to be recognized, and have the right code : they are "thresholded" linear combinations of prototype patterns with similar shapes. The thresholded linear combinations of prototype patterns with non-similar shapes act as "garbage" collectors ; they have a meaningless code.

A more detailed analysis can be presented : 40% of the iterations lead to a prototype state, among which there are less than 1% misclassifications ; 60% of the iterations lead to a thresholded linear combinations of the prototype states, approximately 2/3 of them are well recognized (i.e. have the right code), 1/6 are misclassified and 1/6 unrecognized. The energy of the well recognized thresholded linear combinations is lower than the energy of the other thresholded linear combinations. Thus, the evolution in state space can be interpreted as follows : the *projection rule* generates many low energy stable states near the bottom of the basins of attraction, which can be used for pattern recognition, and also a lot of bumps at the top of the energy barriers, which act as traps for unrecognizable states. Therefore, the thresholded linear combinations of the prototype states too often termed "spurious" states, are used to our advantage : they double the efficiency of the pattern recognition. It has been argued that modifying the present model by equalling the diagonal term of the synaptic matrix to zero increases the bassin of attraction of the prototype states. This is indeed true ; however, a price has to be paid : the synaptic matrix is no longer a projection matrix, so that the pattern recognition capability is degraded. We have tested this idea with the same prototype patterns and we have observed that the number of misclassifications is significantly increased.

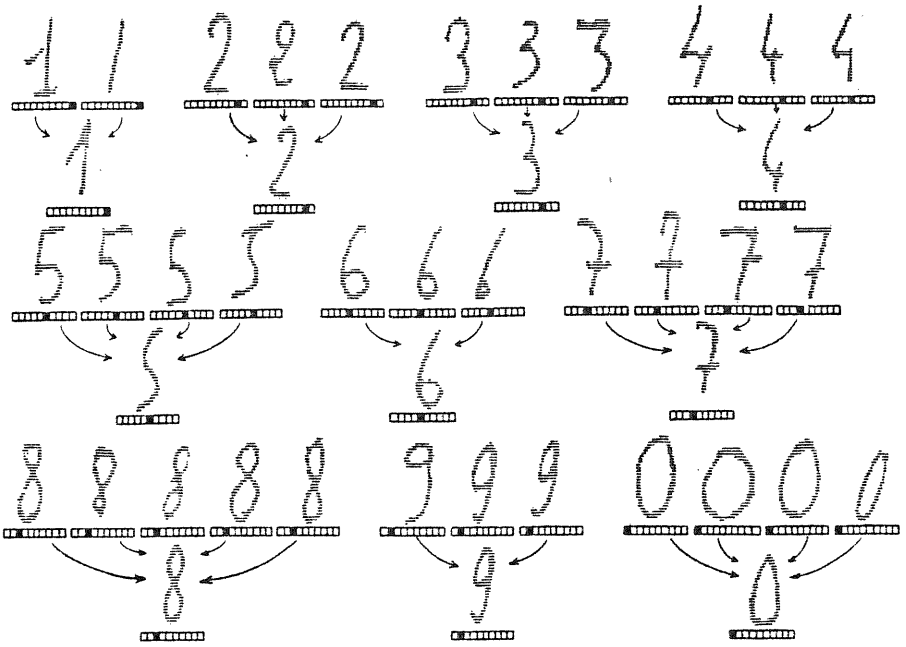


(a)

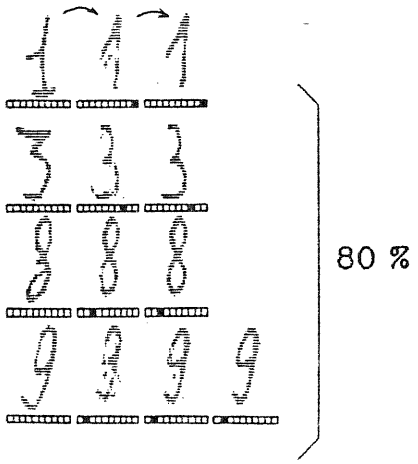


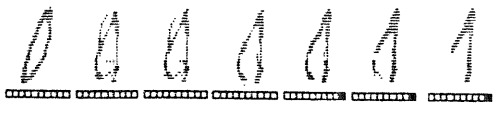
(b)

FIGURE 5 : **Character recognition : *projection rule.***
Number of neurons : n = 1024,
number of prototypes : p = 44.
a. The prototype states.
b. Examples of evolutions.



(a)





(b)

FIGURE 6 : **Character recognition :** ***associating rule.***
Number of neurons : n = 1024,
number of imposed transitions : p = 44.
a. The imposed transitions.
b. Examples of evolutions.

Associating rule.

In order to compare with the previous example, we have stored the same 44 handwritten numerals , with their 10-bit code, using the *associating rule*. The imposed transitions are shown in figure 6.a. The neural network classifier has been tested with the same 250 unknown patterns , with the following results : approximately 80% of the patterns are well recognized, the others are misclassified and only a small percentage are unrecognized. Some examples are shown in figure 6.b. The results can be interpreted as follows : the *associating rule* digs wide basins of attraction for each class of numerals, and very few non imposed attractors appear ; the network almost always reaches one of the imposed fixed points. This can be very suitable for some other classification application but, for character recognition, it is highly desirable to have more unrecognized than misclassified patterns. Therefore, for this application, we shall prefer networks designed with the *projection rule.*

Comparison with Hebb's rule.

The ability of neural networks designed with the above presented rules to store and retrieve information has been compared to that of a network designed after Hebb's rule. The training set is the same as in figure 3.a. No character whatsoever are recognized if Hebb's rule is used, and all the evolutions lead to a single huge attractor.

Comparison with classical methods.

Two other methods of comparable complexity have been tested for the same example: the method of comparison of distances and the optimal linear classifier[23, 24].

The method of comparison of distances consists in computing the Hamming distances $H(\sigma,\sigma^k)$ between the unknown pattern σ and the prototype patterns σ^k. The classification is performed by considering the first and the second nearest neighbors (σ^α and σ^β) of σ, with two thresholds A (absolute) and D (differential). The classification procedure is the following :

i) If $H(\sigma,\sigma^\alpha) > A$, σ is unrecognized ;

ii) If $H(\sigma,\sigma^\alpha) < A$,

- if σ^α and σ^β belong to the same class, σ is attributed to this class ;
- if σ^α and σ^β do not belong to the same class,

- if $H(\sigma,\sigma^{\beta}) - H(\sigma,\sigma^{\alpha}) > D$, σ is attributed to the same class as σ^{α} ;

- if $H(\sigma,\sigma^{\beta}) - H(\sigma,\sigma^{\alpha}) < D$, σ is unrecognized.

Conversely, the optimal linear classifier[23] uses the global information present in the prototype family: like the neural network in the first iteration, it involves the orthogonal projection of the the unknown pattern σ into the subspace spanned by the prototype vectors σ^{k}. More precisely, this classifier computes the product of a (c,n) matrix

$$C = F\,\Sigma^{I},$$

by the vector σ, where c is the number of classes, Σ^{I} is the (p,n) pseudoinverse matrix of Σ, and F a (c,p) matrix such that $F_{ri}=1$ if σ^{i} belongs to class r and zero else.

The classification is performed by considering the two largest components of $C\sigma$, with two thresholds A and D, in a way similar to the method of the comparison of distances. Notice that the decision for the two above classifiers is taken in one step, whereas the decision of neural networks is distributed both in space and in time thus giving them a robustness which is absent in the other methods.

The same set of examples as in figures 5 and 6 has been used to test these two classification processes ; the results are shown in figure 7 together with the results obtained with the neural networks. We have plotted three proportions : proportion of recognition, proportion of misclassification and proportion of unrecognition. The adjustment of the decision thresholds A and D allows to decrease the proportion of misclassification by introducing the possibility that the system refuse to classify some dubious patterns, but, correlatively, the proportion of recognition also decreases. The choice of the thresholds A and D must achieve a compromise (fig. 7.2 and 7.3), and it can be noticed that there is no such degree of freedom for the neural networks. The first a) columns for each method show a good proportion of recognition balanced by a high proportion of misclassification : the *associating rule* is used for the neural network and the differential threshold D is set to zero for the other methods. In the second columns b) the proportion of unrecognition is emphasized : it is achieved, for the neural network, by using the *projection rule* which introduces the "garbage states" ; for the other methods, it is achieved by ajusting the differential threshold D so as to have

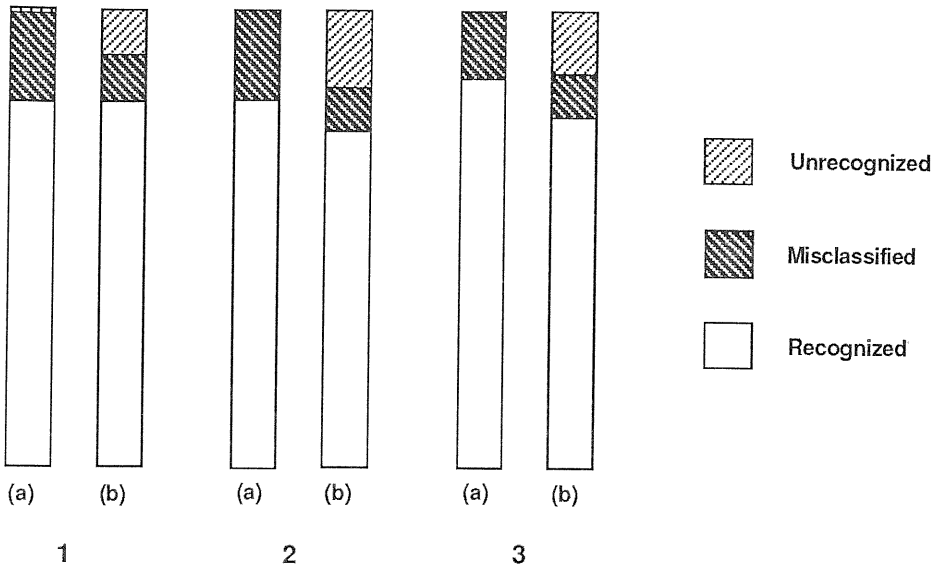


FIGURE 7 : **Character recognition : comparison with other classifiers.**

1.Neural network designed after :
   a) the *associating rule*,
   b) the *projection rule*.
2. Comparison of distances method :
   a) A=130, D=0,
   b) A=130, D=10.
3. Optimal linear classifier :
   a) A=.2, D=0,
   b) A=.2, D=.06.

more conventional, and better established, methods. We have presented some preliminary results which show that neural networks, in their present state, can be competitive ; obviously, much progress is still needed in order to outperform classical approaches, but, considering the youth of this subject, many new developments can be expected.

REFERENCES

[1] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, Science **220**, 671 (1983).

[2] P. Siarry, G. Dreyfus, J. de Physique Lett. **45**, L39 (1984).
P. Siarry, L. Bergonzi, G. Dreyfus, 1er Colloque National sur les circuits à la demande, Grenoble (1985).

[3] D. Rothman, Geophysics **50**, 2784 (1985).

[4] R. G. Paxman, W. E. Smith, H. H. Barrett, J. Nucl. Med. **25**, 700 (1985).

[5] P. Carnevali, L. Coletti, S. Patarnello, IBM J. Res. Dev. **29**, 569 (1985).

[6] S. Geman, D. Geman, IEEE Trans. Pattern Analysis and Machine Intelligence **6**, 721 (1984).

[7] V. Premti, Thèse, Grenoble (1984).

[8] J. P. Uhry, private communication.

[9] R. Sharpe, B. S. Marksjö, J. R. Mitchell, J. R. Crawford, Appl. Math. Modelling **9**, 207 (1985).

[10] J. L. Lutton, E. Bonomi, preprint.

[11] M. Brady, Computer Aided Design **16**, 253 (1985).

[12] M. May, Computer Aided Design **17**, 25 (1985).

[13] E. H. L. Aarts, P. J. M. van Laarhoven, Philips J. Res. **40**, 193 (1985).

[14] E. H. L. Aarts, F. M. J. de Bont, E. H. A. Habers, P. J. M. van Laarhoven, Lecture Notes in Computer Science **210**, 87 (1986).

[15] P. Roussel-Ragot, P. Siarry, G. Dreyfus, 2ème Colloque National sur les circuits à la demande, Grenoble (1986).

[16] S. A. Kravitz, R. A. Rutenbar, Proc IEEE/ACM Design Automation Conference (1986).

almost the same rate of misclassification for the three methods. We can notice that, comparing to case a), the rate of recognition is worse except for the neural network.

Notice that the choice of the prototype patterns and their number per class is still an open problem for all the mentioned methods ; in the present examples, we have used empirical criteria based on the study of the Hamming distances between patterns taken from a set of 250 examples ; this choice is not yet optimized.

4) Conclusion :

We have presented two learning rules for networks of formal neurons :

- the projection rule "digs holes" in state space because it allows to memorize as attractors any set of prototype states;
- the associating rule "digs valleys" in state space because it can impose transitions between states.

We have shown that it is possible, by using directly the pixel representation, to perform character recognition tasks. An important point is that the projection rule generates non-prototype fixed points which are useful : they double the efficiency of the pattern recognition.

Finally, we have compared the networks of formal neurons to other methods of comparable complexity ; the result is that the networks of formal neurons are already competitive, even though the choice of the prototype patterns is not yet optimized.

IV - CONCLUSION :

The engineering applications which have been presented in this paper are widely different in nature. Simulated annealing is more and more routinely used, and is here to stay as an engineering aid for very complex problems ; neural networks are not so mature as far as applications are concerned, but they are rapidly gaining much industrial interest ; one of the issues is, of course, their efficiency as compared to that of

[17] J.J. Hopfield, Proc. Natl. Acad. Sci. USA **79**, 2554 (1982).

[18] L. Personnaz, I. Guyon and G. Dreyfus, J. Phys. Lett. **46**, L-359 (1985).
L. Personnaz, I. Guyon and G. Dreyfus, Phys. Rev. A, to be published.

[19] D.J. Amit, H. Gutfreund and H. Sompolinsky, Phys. Rev. A **32**, 1007 (1985).
I. Kanter and H. Sompolinsky, "Second Generation Neural Networks", present book.

[20] A. Albert, "Regression and the Moore-Penrose Pseudoinverse" (Academic Press, 1972).

[21] L. Personnaz, I. Guyon, G. Dreyfus and G. Toulouse, J. Stat. Phys. **43**, 411 (1986).

[22] D. Hebb, "The organization of behavior" (Wiley, 1949).

[23] T. Kohonen, "Self organization and associative memories" (Springer, 1984).

[24] See for instance : R.O. Duda and P.E. Hart, "Pattern recognition and scene analysis" (Wiley, 1973).

398 = vakat