TREATING WEIGHTS AS DYNAMICAL VARIABLES

-- A NEW APPROACH TO NEURODYNAMICS --

U. RAMACHER M. WESSELING

SIEMENS AG, Corp. R&D
Otto-Hahn-Ring 6, 8000 Munich 83, F.R. GERMANY

ABSTRACT

The recall and learning dynamics of Artificial Neural Networks are described by means of a partial differential equation (PDE) that may incorporate weights either as parameters or variables. In the case that weights are interpreted as variables, a new type of neurodynamics is discovered: weights have to obey second order differential equations called learning laws. Experiments on the association of time-varying patterns indicate the superiority of the learning law over the known types of learning rules. It is also shown that a single first-order Hamilton-Jacobi 'parametric' PDE suffices to derive the various neurodynamical paradigms used today [1-5].

INTRODUCTION

A popular approach to the description of the learning dynamics of neural networks looks at the weights as being parameters that have to be fitted according to some learning rule [1-6]. Moreover, weights are assumed to be time-invariant after learning. Since neural networks with their thousands or millions of weights run the danger of, firstly, over-fitting the experiments and, secondly, the use of parameters tends to darken the actual dynamics of the system, we suggest to treat weights in the same manner as neurons, ie as dynamical variables of the system. Hence, 'dynamical' weights have to obey an ODE, like the neurons.

THE HAMILTONIAN CONCEPT AND ITS GEOMETRICAL INTERPRETATION

The basic variables for the dynamical description of a neural network are considered to be the time t, the neural states y and the weight states W. The change of the states is influenced by two sources: external inputs and physical implementation. In this paper, dynamical effects caused by the physical implementation are not considered (see [7], for example) . The external inputs may comprise reference signals for learning as well as actual input signals. Both types of inputs form the (generally time-varying) boundary conditions which control any change in the state space of the neural net. Besides, the action of the system invokes a set of observables J(t) each of which can be considered as the collective result of the states and the boundary conditions: J(t): = J(t,y(t),W(t)).

Figure 1 depicts a trajectory J(t,y(t),W(t)). Changing the boundary conditions (for instance, by means of a new pattern input) or the initial states of the neural network one obtains a new trajectory. Imagining that all possible boundary conditions were experienced by the network, one obtains a surface J: = J(t,y,W) that represents the complete information about the dynamics of the neural network. Since beside t, y, W and J the partial derivatives of J with respect to all its variables carry the knowledge of how the surface J(t,y,W) is formed it is natural to request that the surface J must obey an equation

$$D(t, y, W, J, \partial J/\partial t, \partial J/\partial y, \partial J/\partial W) = 0$$

where D is some operator function. We restrict ourselves to a particularly simple operator D:

0-7803-0559-0/92 $3.00 © 1992 IEEE

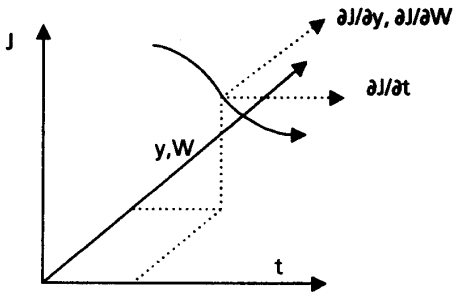

Figure 1 Trajectory of an observable over the state space

$$\frac{\partial J}{\partial t} + H\left(t, y, \Delta, W, M\right) = 0 \quad , \quad with \ \Delta_i := \frac{\partial J}{\partial y_i} \ , \ M_{ij} := \frac{\partial J}{\partial W_{ij}} \quad . \tag{1}$$

In the theory of partial differential equations it is shown [8,9] that Equation (1), which is called Hamilton-Jacobi, gives rise to a fundamental set of ordinary differential equations, the so-called characteristic equations:

$$\frac{dy_i}{dt} = \frac{\partial H}{\partial \Delta_i} \ , \ \frac{d\Delta_i}{dt} = -\frac{\partial H}{\partial y_i} \ , \ \frac{dW_{ij}}{dt} = \frac{\partial H}{\partial M_{ij}} \ , \ \frac{dM_{ij}}{dt} = -\frac{\partial H}{\partial W_{ij}} \ , \tag{2a}$$

$$\frac{dJ}{dt} = \frac{\partial J}{\partial t} + \sum_j \Delta_j(t) \cdot \frac{\partial H}{\partial \Delta_j}(t) + \sum_{ij} M_{ij}(t) \cdot \frac{\partial H}{\partial M_{ij}}(t) \ , \tag{2b}$$

Once the solutions for the characteristic equations (2a) are known, the trajectory J(t,y(t),W(t)) can be determined by direct integration of the Equation (2b). The corresponding surface J = J(t,y,W) is generated by the manifold of all solutions of (2a), obtained by varying the initial states of the system and the boundary conditions, ie by confronting the neural network with sufficiently many input as well as reference patterns.

Integration of (2b) yields:

$$J(t) = J(t_i) + \int_{t_i}^{t} \left(\sum_j \Delta_j(\tau) \cdot \frac{dy_j}{d\tau} + \sum_{ij} M_{ij}(\tau) \cdot \frac{dW_{ij}}{d\tau} - H(\tau) \right) d\tau \tag{2c}$$

Then, searching for an extremal trajectory between two points of the state space yields [10] :

$$\delta J = \left\{ \sum_k \Delta_k(\tau) \cdot \delta y_k(\tau) + \sum_{ij} M_{ij}(\tau) \cdot \delta W_{ij}(\tau) - \frac{\partial H}{\partial \tau} \cdot \delta\tau \right\} \Bigg|_{\tau = t_f}$$

$$+ \int_{t_i}^{t_f} dt \left[\left(\frac{dH}{dt} - \frac{\partial H}{\partial t} \right) \cdot \delta t + \sum_k \left[\left(-\frac{d\Delta_k}{dt} - \frac{\partial H}{\partial y_k} \right) \cdot \delta y_k + \left(\frac{dy_k}{dt} - \frac{\partial H}{\partial \Delta_k} \right) \cdot \delta \Delta_k \right] +$$

$$\sum_{ij} \left[\left(-\frac{dM_{ij}}{dt} - \frac{\partial H}{\partial W_{ij}} \right) \cdot \delta W_{ij} + \left(\frac{d}{dt} W_{ij} - \frac{\partial H}{\partial M_{ij}} \right) \cdot \delta M_{ij} \right] \right] \quad . \tag{2d}$$

The most general set of necessary conditions for an extremal trajectory $J(t_f)$ therefore reads:

$$\delta J = 0 \Rightarrow \left[\forall \ i,j : 0 = \Delta_i(t_f) \ , \ 0 = M_{ij}(t_f) \ ; \ 0 = H(t_f) \right] \ . \tag{3a}$$

Obviously, any choice of a Hamiltonian leads to an extremal trajectory J(t). This underlines clearly the importance of the construction of the Hamiltonian that allows to learn. It is noted also that recall and learning have to happen simultaneously.

Remark: If weights are not conceived as variables but as parameters, the type of neurodynamics envisaged by Optimal control [5] can be derived. Then the conjugate variable M associated with W would not exist, and therefore the characteristic equations be reduced to equations for y, Δ and J. Consequently, the derivative ∂H/∂W would remain unbalanced by a dynamic term, but give rise to an extra term in (3a):

$$\forall\, t_i \le t \le t_f \;:\; \frac{\partial H}{\partial W_{ij}}(t) = 0 \;. \tag{3b}$$

Any method producing weights that satisfy (3b) constitutes a learning rule (for details see [11]).

EXAMPLE OF A HAMILTONIAN THAT GENERATES A LEARNING LAW

We assume a recursive direct decomposition of the weight functions according to what has been and what remains to be learned during an epoch of width T:

$$W(t) = W^T(t) + w(t) \;,\; with \;\; W^T(t) := W(t-T) \;. \tag{4a}$$

Consequently, we arrive at a similar decomposition for the conjugate variables of the weights:

$$M(t) := M^T(t) + m(t) \;,\; with \;\; M^T := \frac{\partial J}{\partial W^T} \;,\; m := \frac{\partial J}{\partial w} \;. \tag{4b}$$

Let us consider the Hamiltonian:

$$H(t) = \sum_k \Delta_k(t) \cdot \; F_k(t) + \frac{1}{\omega} \sum_{k,l} M^T_{kl}(t) \cdot \; m_{kl}(t) + E\Big(y(t), W_T(t)\Big) - \sum_{k,l} \frac{dm_{kl}}{dt}(t) \cdot \; w_{kl}(t) \;, \tag{5}$$

with

$$F_k(t) := -y_k(t) + f_k\Big(\sum_{j=-1}^{N} W^T_{kj}(t) \cdot \; y_j(t) \Big) \;.$$

The Hamiltonian (5) yields the characteristic Equations:

$$\frac{dy_i}{dt} = F_i(t) \;,\quad \frac{d\Delta_i}{dt}(t) = \Delta_i(t) - \sum_j \Delta_j(t) \cdot \; \dot{f}_j(t) \cdot \; W^T_{ji}(t) - \frac{\partial E}{\partial y_i}(t) \;, \tag{6a}$$

$$\frac{dM^T_{ij}}{dt} = -\frac{\partial H}{\partial W^T_{ij}} = -\Delta_i(t) \cdot \; \dot{f}_i(t) \cdot \; y_j(t) - \frac{\partial E}{\partial W^T_{ij}}(t) \;,\quad \frac{dm_{ij}}{dt} = -\frac{\partial H}{\partial w_{ij}} = \frac{dm_{ij}}{dt} \;, \tag{6b}$$

$$\frac{dW^T_{ij}}{dt} = \frac{\partial H}{\partial M^T_{ij}} = \frac{1}{\omega} \cdot \; m_{ij}(t) \;,\quad \frac{dw_{ij}}{dt} = \frac{\partial H}{\partial m_{ij}} = \frac{1}{\omega} \cdot \; M^T_{ij}(t) \;. \tag{6c}$$

The Equations (17c-d) can be combined to yield second order ODEs for the weights:

$$\frac{d^2 W^T_{ij}}{dt^2} = \frac{d^2 W^T_{ij}}{dt^2} \;,\quad \frac{d^2 w_{ij}}{dt^2}(t) = -\frac{1}{\omega} \cdot \; \Big[\Delta_i(t) \cdot \; \dot{f}_i(t) \cdot \; y_j(t) + \frac{\partial E}{\partial W^T_{ij}}(t) \Big] \;. \tag{6d}$$

It follows that the ODEs for the variables y, Δ, w and M^T accumulate new knowledge, whereas the remaning ones for W^T and m contain the knowledge already learned (ie the latter ones need not be computed).

A qualitative understanding of the Equations (6a-c) is obtained by assuming a constant reference output r and any feedforward network. For simplicity the error function $E(t) = -1/2\,(y(t)-r)^2$ is chosen. If the reference r is set to 0 the weights w are expected to decrease in time. Now, the error $\partial E/\partial y = -y$ is never positive, since f is the sigmoidal function. Hence, by integrating the Δ-equation from $\Delta(T) = 0$, Δ must be negative for the output layer. This implies M^T to be negative, too. It follows that ω has to be positive in order that the weights w decrease.

In the next time epoch of length T, the 'new' weights w are added to the 'old' weights W^T, in accordance with the decomposition (4a) of the weights. Repeating the integration of the Equations (6a-c) one arrives at another set of 'new' weights which are smaller in magnitude than the first set, because of the error correction induced by the first update. Similarly, the variables Δ and M^T of the hidden layer are forced by the Δ of the output layer to change in such a manner that the error E is diminuished.

In order to test the Equations (6a-c) and compare their performance to the ones obtained with learning rules, a pattern association task was performed. Three time-varying inputs were to be associated to three time-varying reference patterns (see Figure 3). The network chosen is composed of 1 input, 15 hidden and 1 output neuron, and denoted as R1-15-1. The hidden layer is fully connected. The error function for the output of the network was $E(t) = -1/2\,(y(t)-r(t))^2$.

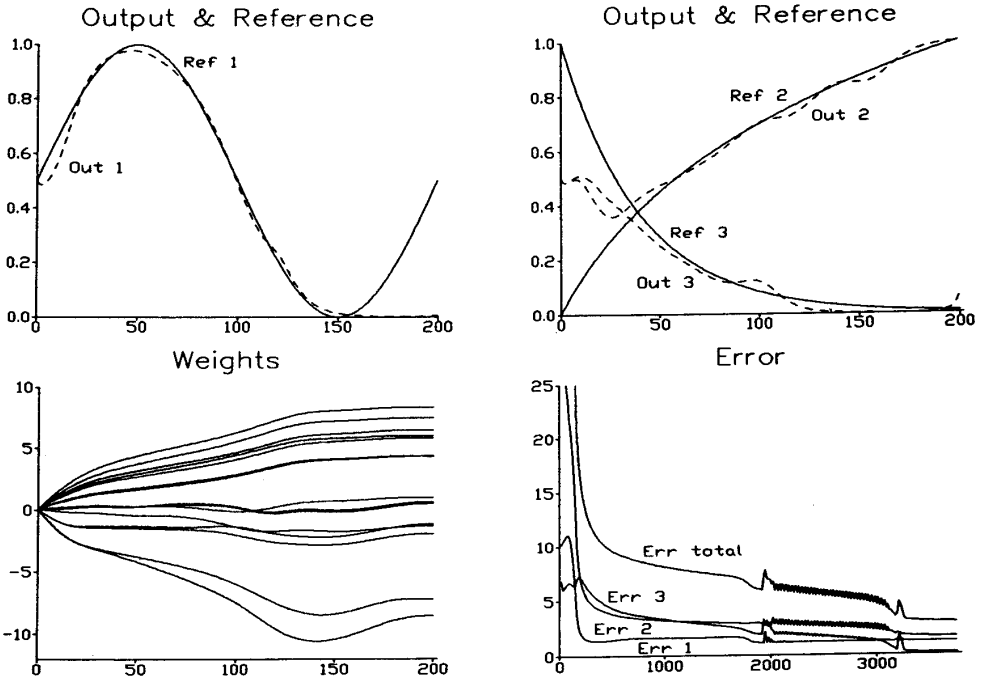

Figure 2 Results for R1-15-1 and the Equations 6a-c ($\omega^{-1} = 0.1$, step size = 0.1)
Upper figures: output of the net after 4000 updates; Lower figures: Weight functions of the output neuron after 4000 updates (horizontal units = # of samples); Accumulated total and partial errors (horizontal unit = # of updates)

Figure 2 shows the results obtained for R1-15-1, with $\omega^{-1} = 0.1$ and constant step size of 0.1 . The equations were found to converge to zero update changes; more precisely: $n \to \infty \Rightarrow \Delta(n \cdot T + \tau) \to 0$, for all $0 \le \tau \le T$. Note that in the limit one obtains the relation:

$$W(t) = W(t-T)$$

telling that the weights functions are either constant or time-varying. In any case, they are

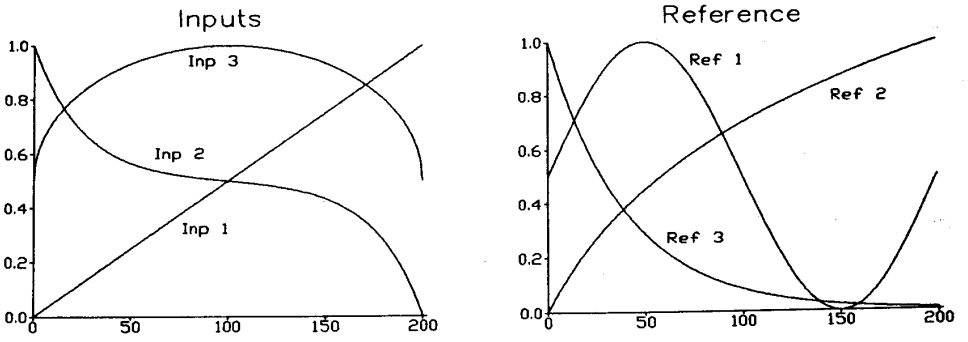

Figure 3 Input and reference patterns

repeated periodically unless a pattern is presented that restarts the learning process.

Figure 4 presents the results obtained for R1-15-1 when using the following equations:

$$\frac{d}{dt}y_i(t,s) = -y_i(t,s) + f_i\left(\sum_j W_{ij}(t,s)\cdot\ y_j(t,s)\right), \tag{7a}$$

$$\frac{d\Delta_i}{dt}(t,s) = \Delta_i(t,s) - \sum_j \Delta_j(t,s)\cdot\ \dot{f}_j(t,s)\cdot\ W_{ji}(t,s) - \frac{\partial E}{\partial y_i}(t,s)\ ,\ \Delta_i(T,s)=0\ . \tag{7b}$$

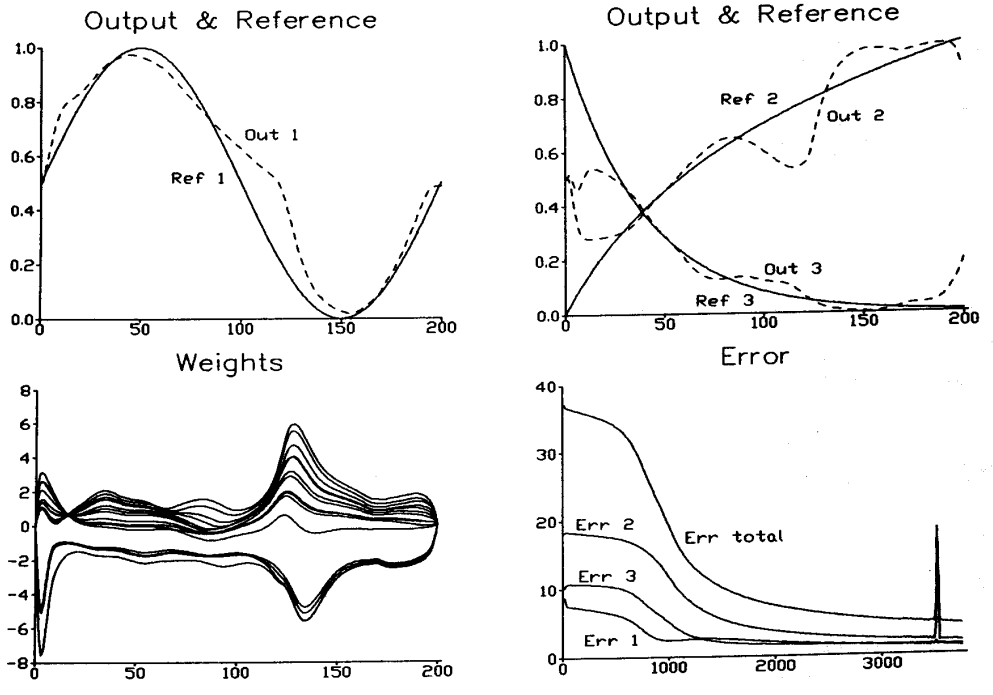

Figure 4 Results for R1-15-1 in case of the Equations 7a-c (α = 10, step size = 0.1)
Upper figures: Output of the net after 4000 updates. Lower figures: Accumulated weight functions of the output neuron after 4000 updates (horizontal units = # of samples); Accumulated total & partial errors (horizontal unit = # of updates).

$$\frac{dW_{ij}}{ds}(t,s) = \alpha \cdot \left[\Delta_i(t,s) \cdot f_i'(t,s) \cdot y_j(t,s) + \frac{\partial E}{\partial W_{ij}}(t,s) \right] . \qquad (7c)$$

Equations (7a-c) are the gradient version of the characteristic equations of the Hamiltonian

$$h(t) = \sum_j \Delta_j(t) \cdot F_j\left(y(t); W(t) \right) + E\left(t, y(t); W(t) \right) , \qquad (7d)$$

which treats weights as time-varying parameters (see relation (3b) or [11,5]).

In the case that weights are restricted to constant parameter functions the condition (3b) as well as Equation (7c) have to be integrated over time [11]. The combined result reads:

$$\frac{dW_{ij}}{ds}(s) := \frac{1}{T}\int_0^T \frac{\partial h}{\partial W_{ij}}(t,s)\, dt = \frac{\alpha}{T}\int_0^T \left[\Delta_i(t,s) \cdot f_i'(t,s) \cdot y_j(t,s) + \frac{\partial E}{\partial W_{ij}}(t,s) \right] dt . \qquad (8)$$

A relation similar to (10) was obtained by Pearlmutter for a special cost function [4] . Figure 5 depicts corresponding results.

It is to be noted here that the variable 's' has no interpretation as a dynamical process run by the network. This is in contrast to the fully dynamical case which has a single time t for recall as well as learning. Hence, we may differentiate between learning rules fulfilling (3b) and learning laws similar to (6d).

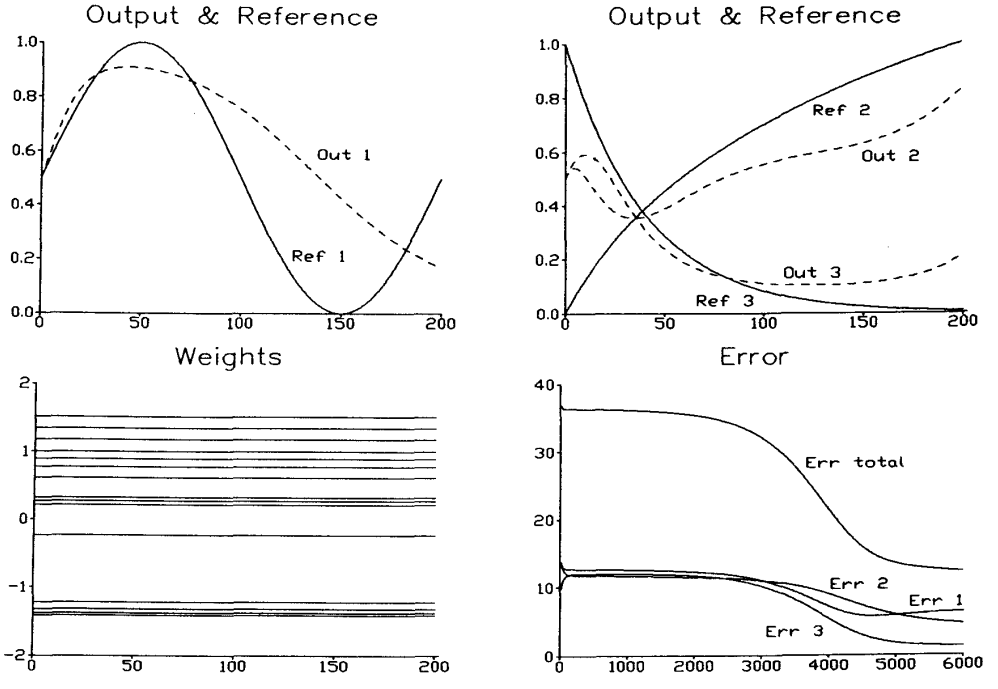

Figure 5 Results for R1-15-1 and the Equations 7a-b,8 (α/T = 0.1, step size = 0.1)
Upper figures: output of the net after 6000 updates. Lower figures: Accumulated weight functions of the output neuron after 6000 updates (horizontal units = # of samples); Accumulated total & partial errors (horizontal unit = # of updates)

COMPARING THE FIGURES 2,4,5: It is clearly seen that Equations (6a-c) give the fastest decline of the error functions in terms of updates as well as computation time. Further, fitting with constant weights lasts considerably longer in terms of equal error. In passing we note that the recurrent network R1-15-1 was found always to work better than the feedforward version of R1-15-1.

CONCLUSIONS

We have described a neural network as being a dynamical system. Its observables are introduced as solutions of a partial differential equation of Hamilton-Jacobi type. The observables are functions of time, the neural states and the weight states. The dynamics of an observable is interpreted as a surface over the state space of the neural net which is generated by all its admissible trajectories.

If the weights are treated as parameters, the only dynamical process that actually will be run by the network is the recall process. It is shown that the known types of learning dynamics can be reproduced in this framework, and that weights are generally time-varying during recall. This feature is shown to account for much faster learning as compared to weights being constant for each learning epoch.

If, on the other hand, weights are conceived as variables, a fully dynamical new picture of recall and learning results. Neurons as well as weights obey differential equations, ie recall and learning are dynamical processes of the neural net. It is important to note that the differential equations to be obeyed by neurons and weights, respectively, constitute half of the characteristic equations associated with the Hamilton-Jacobi equation. The remaining equations are obeyed by the conjugate variables of the neurons and weights, respectively. These latter are mediating the interaction between neurons and weights. Comparing the fully dynamical concept with the optimization approach (that exploits techniques from Optimization Theory, Control Theory or Dynamic Programming [12]), the dynamical concept was found to learn much faster than the optimization concept.

ACKNOWLEDGEMENT: We are particular grateful to B. Schürmann for discussing with us the Hamiltonian concept and its interpretation with respect to neural networks.

REFERENCES

[1] D. Rumelhart, G. Hinton, and R. Williams: "Learning Internal Representations by Error Propagation". In PARALLEL DISTRIBUTED PROCESSING: EXPLORATIONS IN THE MICRO-STRUCTURE OF COGNITION, Vol I, MIT Press 1986

[2] L.B. Almeida: "Backpropagation In Non-Feedforward Networks". IEEE INT. CONF. ON NEURAL NETWORKS, SAN DIEGO, PP. II-609, 1987

[3] F.J. PINEDA: "Generalization Of Backpropagation To Recurrent Neural Networks". PHYS. REV. LETTERS 59, pp. 2229, 1987

[4] B.A. Pearlmutter: "Learning State Space Trajectories In Recurrent Neural Networks". NEURAL COMPUTATION 1, pp. 263, 1989

[5] O. Farotimi, A. Dembo, and T. Kailath: "A General Weight Matrix Formulation Using Optimal Control". IEEE TRANSACTION ON NEURAL NETWORKS, May 1991, Vol 2 No. 3, pp. 378

[6] U. Ramacher, B.Schürmann: "Unified Description of Neural Algorithms For Time-Independent Pattern Recognition", in: VLSI DESIGN OF NEURAL NETWORKS, Kluwer Acad. Publishers, 1991

[7] L.O. Chua, L. Yang: "Cellular Neural Networks: Theory and Applications". IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS, Vol. 35, No. 10, pp. 1257-1290, Oct. 1988

[8] E. Kamke, PARTIAL DIFFERENTIAL EQUATIONS, B.G. Teubner, Stuttgart 1977

[9] F. Verhulst, NONLINEAR DIFFERENTIAL EQUATIONS AND DYNAMICAL SYSTEMS, Springer 1985

[10] A.P. Sage and C.C. White, OPTIMUM SYSTEMS CONTROL, Prentice-Hall, 1977

[11] U. Ramacher, M. Wesseling: "Hamiltonian Approach to Neural Network Dynamics", Proc. IJCNN-91, Singapore, Vol. 3, pp.1930-1936, Nov. 1991

[12] L.S. Pontrijagin et al. , THE MATHEMATICAL THEORY OF OPTIMAL PROCESSES, Wiley, 1962