Asymptotically Stable Recurrent Neural Networks and their Use, an Overview

Nikitas J. Dimopoulos, Stephen Neville

Department of Electrical and Computer Engineering,
University of Victoria,
PO BOX 3055, Victoria, BC, V8W 3P6
CANADA

Abstract –In this work we present a class of recurrent networks which are asymptotically stable. For these networks, we discuss their similarity with certain structures in the central nervous system, and prove that if an interconnection pattern that does not allow excitatory feedback is used, then the resulting recurrent neural network is stable. We introduce a training methodology for networks belonging to this class, and use it to train networks that successfully identify a number nonlinear systems.

I. Introduction

Neurons and their interconnections constitute the Nervous System in living organisms. The Nervous System by design and/or training implements functionality that enables the organism to survive in its environment. It is worth therefore studying this functionality in relation to the structure and organization of the Nervous System, in the hope that artificial analogs could be devised possessing interesting properties and perhaps mimicking some of the functionality found in Nature.

In what follows, we shall present some general observations regarding the organization and interconnection of neurons in the Nervous System. We shall present the specialization of these principles in the structure of a particular part of the Nervous System (the Cerebellum) and we shall see that such a structure guarantees the stability of the part. Further, we shall present artificial analogs in the form of neural networks and a training method that allows these analogs to *learn* the behavior of an arbitrary dynamical system. Again, structures which were found in Nature, shall prove crucial in training. Finally, we shall present examples where these artificial analogs have been applied.

A. Physiological Principles

In examining the structure of Nervous Systems, one can observe that:

Neurons belong to physiologically and morphologically distinct groups. Neurons within a group have similar properties (e.g. are all ihibitory, all are connected in a similar manner with neurons of another group). and

Neurons are only connected locally. (A neuron does not normally affect nor is affected by every neuron in the network).

We understand the above general statements as the *macroscopic microscopic* connectivity principles [8].

An example structure, which has been used extensively is the cerebellum. This structure is found posterior to the cerebrum and it is believed that it is used to make movement smooth. Patients with trauma in their cerebellum posses the ability to move their limbs towards a desired position, but the movement is jerky and erratic.

A section of the cerebellum can be seen in Figure 1. This structure includes neurons belonging to four different classes, namely Purkinje, Basket, Golgi and Granule cells. The three first classes comprise inhibitory neurons, while the only excitatory neurons are the Granule cells. Input is provided via the Mossy and Climbing fibers, while the axons of the Purkinje cells are the only afferent (output) pathways.

It is noted that these neurons are connected in a particular fashion. For example, the axons of the granule cells become elongated and are arranged in parallel to each other forming the Parallel Fibers. Neurons from all the classes but granule cells, receive input from the Parallel Fibers. The absence of self loops (granule cells to granule cells via the Parallel Fibers) contributes to the stability of the structure, as we shall formally prove in the subsequent. Additionally, the arborizations of both the dendrites and the axons are limited, and thus neurons are affected by neurons which are in their immediate vicinity (microscopic connectivity principle).

Diagrammatically, one can represent the interconnections in a Nervous System by presenting the way that the composing the structure neural classes connect. The interconnect for the cerebellum is presented in Figure 2. One can verify the absence of any excitatory feedbacks.

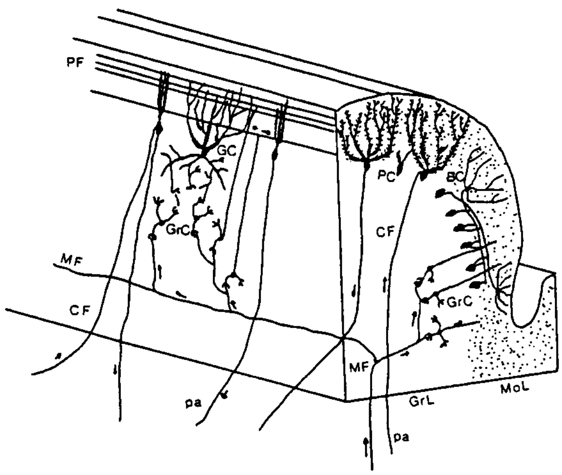

Fig. 1. The structure of the cerebellum. PC: Purkinje Cells, GrC: Granule Cells, GC: Golgi Cells, BC: Basket Cells, PF: Parallel Fibers, MF: Mossy Fibers, CF: Climbing Fibers, pa: axons of the Purkinje cells.

II. Recurrent Neural Networks, Stability And Learning

A. *Recurrent Neural Networks*

Neural networks with structures which obey the two connectivity principles discussed earlier are described by the differential equation [8].

$$\dot{O} = -TO + Wf(O) + b \tag{1}$$

In (1), there are N neurons divided into k classes, and

$$O = \begin{bmatrix} O_1 & O_2 & \cdots & O_k \end{bmatrix} = \begin{bmatrix} o_1 & o_2 & \cdots & o_N \end{bmatrix}$$ is the state of the neural network, $$W = \begin{bmatrix} W_{11} & W_{12} & \cdots & W_{1k} \\ \cdots & & & \\ W_{k1} & W_{k2} & \cdots & W_{kk} \end{bmatrix}$$

is the network connectivity matrix, $T = \mathrm{diag}(\tau_i)$ is the diagonal matrix of neural relaxation constants, b is the input to the neural network, and $f(O)$ belongs to the class of so-called neuromime functions, which are positive monotonically non-decreasing satisfying a Lipschitz condition and $\exists \theta \in R^N$ such that $f(\theta) = 0$.

The functions $f(O)$ represent the nonlinearity of the hillock, and define how the membrane excitation is translated to a train of action potentials. A commonly used function is the sigmoid, while the class of neuromime functions is more general and includes in addition to the sigmoids, piecewise linear and continuous functions for example.

The structure of the network is reflected in the structure of the connectivity matrix W. Each submatrix W_{ij} represents the interconnection weights between class i and class j.

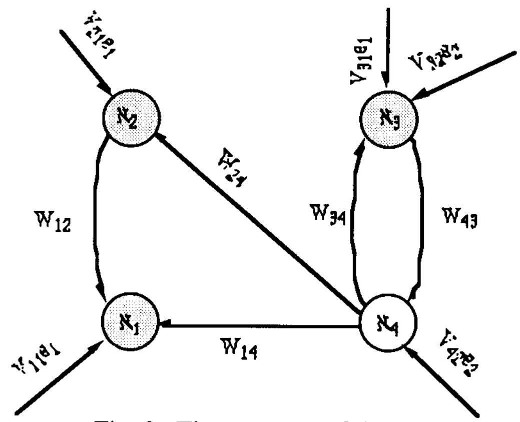

Fig. 2. The structure of the cerebellum.

$\mathcal{N}_1$ Purkinje (Inhibitory)	$\mathcal{N}_2$ Basket (Inhibitory)
$\mathcal{N}_3$ Golgi (Inhibitory)	$\mathcal{N}_4$ granule (Excitatory)
e_1 Climbing fibers	e_2 mossy fibers

B. *Asymptotically Stable Recurrent Neural Networks*

The condition on W that guarantees asymptotic behavior is that it must contain all of its positive entries on one side of the main diagonal. In the subsequent, we analyze the stability of a small example structure, the complete analysis can be found in [8].

The system

$$T_1\dot{O}_1 + O_1 = f_1(O_2)$$
$$T_2\dot{O}_2 + O_2 = -f_2(O_1) \tag{2}$$

with T_1 and T_2 diagonal positive matrices and $f_i(0) = 0$, is asymptotically stable in the large with solutions which are bounded from above and below by functions of the form $e^{-At}P(t)$ where $P(t)$ is a vector polynomial in t. The connectivity matrix corresponding to (2) is

$W = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$ and it obeys the condition of having all its positive entries on the same side of the diagonal.

The solution can be written as

$$O_1 = e^{-T_1^{-1}t}O_{10} + e^{-T_1^{-1}t}\int_0^t e^{-T_1^{-1}s}T_1^{-1}f_1[O_2(s)]\,ds$$

$$O_2 = e^{-T_2^{-1}t} O_{20} + e^{-T_2^{-1}t} \int_0^t e^{-T_2^{-1}s} T_2^{-1} f_2 [O_1 (s)] ds$$

Thus $O_1 \geq e^{-T_1^{-1}t} O_{10}$ and $O_2 \leq e^{-T_2^{-1}t} O_{20}$.

Substituting in the original,

$$O_1 \leq e^{-T_1^{-1}t} O_{10} + e^{-T_1^{-1}t} \int_0^t e^{-T_1^{-1}s} T_1^{-1} f_1 \left[e^{-T_2^{-1}s} O_{20} \right] ds$$

Since $f_1(\bullet)$ is non-negative and non-decreasing

$$O_1 \leq e^{-T_1^{-1}t} O_{10} + e^{-\tilde{T}t} \tilde{P}_1 (t) \quad \text{or}$$

$$O_1 \leq e^{-Tt} \{ |O_{10}| + |\tilde{P}_1 (t)| \}$$

where $T = min(T_1^{-1}, \tilde{T})$

A similar procedure is followed for O_2 and for the general case.

This gives an easy way to check whether a neural network is stable. For instance, the neural network shown in Figure 2. is stable provided that the connection weights in submatrices W_{23} and W_{34} are non-positive (i.e. inhibitory).

This result is extremely useful in the area of identification and control. A most important feature of a controller or model is that it must be stable. This is accomplished by ensuring that the structural condition on the connectivity matrix W that guarantees stability, is maintained.

C. *Parameter Adjustment In Stable Neural Networks*

This section discusses a method for adjusting the weights and other parameters of neural networks that are stable in the sense described in Section . The general approach that is used here is to define some a criterion and then adjust the parameters in a direction that will decrease this cost. In this sense the technique is similar to linear recursive adaptive methods [4] and to classical back propagation [5]. However, since the stable neural networks described in Section have certain restrictions on the polarity of the connection of classes, a straightforward gradient adjustment is not possible. A solution for this is also presented here.

D. *Gradient of Cost Function*

The general equation for calculating the behavior of the class of neural networks of interest here is

$$\dot{O} = -TO + Wf(O) + b \tag{3}$$

using the same notation introduced in section 2. One possible criterion for measuring the performance is the quadratic cost function

$$J(e) = 1/2(O-O_d)^T A (O-O_d) = 1/2 e^T A e \tag{4}$$

where O_d is the desired state of the neural network. Matrix A is used to eliminate from the cost any neurons whose state is not crucial. A is a diagonal matrix with ones corresponding to output neurons and zero's elsewhere. As in other recursive adaptive methods [3],[9], parameters θ in the neural network are adjusted along the negative gradient of this cost, i.e., $\frac{d\theta}{dt} = -\eta \frac{\partial J}{\partial \theta}$. The chain rule for differentiation is used to allow for the calculation of this gradient for parameters associated with neuron j:

$$\frac{\partial J}{\partial \theta} = \frac{\partial J}{\partial o_j} \frac{\partial o_j}{\partial \theta} = \gamma_j \frac{\partial o_j}{\partial \theta} \tag{5}$$

The notation γ_j is used to denote the derivative of the cost with respect to the activation of neuron j. If neuron j is an output neuron, this derivative is simply

$$\gamma_j = o_j - o_{d_j}. \tag{6}$$

In a manner analogous to traditional back propagation of the error [9], this gradient may be calculated for units that are not output neurons by using the values of the gradient in all the neurons k that have neuron j as inputs:

$$\gamma_j = \sum_k \gamma_k \frac{\partial o_k}{\partial o_j} = \sum_k \gamma_k \Delta_{kj} \tag{7}$$

Here, the notation Δ_{kj} has been introduced to represent the partial derivative $\partial o_k / \partial o_j$. To calculate Δ_{kj}, it is necessary to use the differential equation which defines the behavior of the neural network. Rewriting (4) specifically for neuron k, and using the operator D to represent differentiation results in

$$(\tau_k + D) o_k = \sum_j w_{kj} f(o_j) + b_k \tag{8}$$

Differentiating (8) with respect to o_j results in

$$\dot{\Delta}_{kj} = (-\tau_k) \Delta_{kj} + w_{kj} f'(o_j). \tag{9}$$

All the derivatives required in (5) to adjust a parameter θ have now been obtained, except for the derivative $\partial o_j / \partial \theta$. The next section discusses the case when θ is a connecting weight. A similar technique can be used to obtain a formulae for adjusting any of the other variables that parameterize the neural network such as the relaxation constant τ or parameters of the activation function $f(\,)$ [6],[7].

E. Weight Adjustment

Let θ represent a connecting weight w_{ji} which connects neuron i (input) to neuron j. Use the notation $\xi_{ji} = \partial o_j / \partial w_{ji}$. Differentiating (10) with respect to w_{ji}, the differential equation for ξ_{ji} is obtained:

$$\dot{\xi}_{ji} = -\tau_j \xi_{ji} + f(o_i) \tag{10}$$

Using this equation and the results of the previous section, equation (7) may now be written as

$$\frac{dw_{ji}}{dt} = -\eta \gamma_j \xi_{ji} \tag{11}$$

with γ_j calculated using (6) or (7) as appropriate.

F. Weight Clamping

Section describes a class of neural networks that are asymptotically stable. This condition is guaranteed provided that the connectivity matrix W has all of its positive entries on one side of the diagonal. However, (11) gives a formula for adjusting the connection weights that may violate this condition. To combat this, it is necessary to check the polarity of certain crucial weights after each weight adjustment. For instance, if the weights labeled W_{12} and W_{43} in Figure 2 are guaranteed to be non-positive, then the neural network will be stable. Thus after any weight in W_{12} and W_{43} is adjusted using (11), it should be checked to ensure that it is not positive. If it is positive, then it is clamped at 0. This ensures that inhibitory connections remain inhibitory throughout the training procedure.

III. IDENTIFICATION

The term identification is used in this section to refer to the process of developing a model of an unknown system by observing its input/output behavior [3],[9].

This section uses the results of the previous section to identify some unknown systems. A suitable neural network architecture is proposed and some motivation for this configuration is given.

A. Identification Architecture

Consider the simple nonlinear system described by the relation

$$\dot{y} = u - \frac{y}{1 + 4y^2} \tag{12}$$

If y remains relatively constant near some value y_{ss}, then this system can be approximated by a first order linear system that has a pole at $(1 + 4y_{ss}^2)^{-1}$. If y varies from this value significantly, then the 'pole' can be thought of as roving in some sense. Although this is not an exact description of the behavior of the system, it does illustrate one of the more common types of nonlinearity which is encountered in real systems.

To take advantage of this type of nonlinearity, the architecture of Figure 3. is proposed for general system identification. Labels I and O refer to the input and output of the system, and $\mathcal{N}_1$ and $\mathcal{N}_2$ to two classes of neural networks.

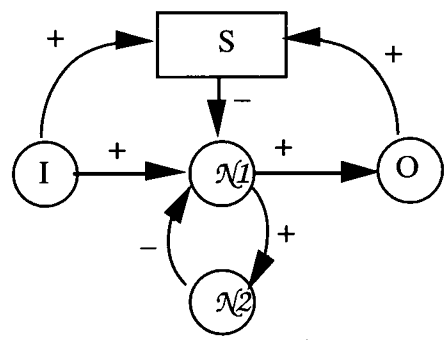

Fig. 3. An Architecture for System Identification

The block marked S is a special connection of classes called the "scheduler class". The idea of this class is that it schedules which neurons will be active and when, thereby emulating the movement of the "pole" for large variations of the state variable or input. Neurons in the scheduler class have a "peaked" response as shown in Figure 4. Each neuron in the class has a peak at ρ that occurs at a different value. Figure 3. shows that the scheduler class receives input from I and O. Depending on the values of the input and output, different neurons in $\mathcal{N}_1$ and $\mathcal{N}_2$ will be active.

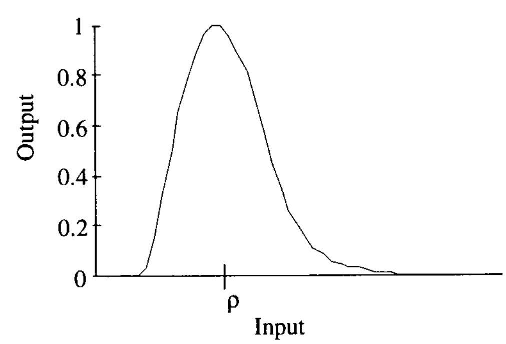

Fig. 4. Response of Scheduler Neurons

This allows the neural network to take advantage of the type of nonlinearity discussed above. The example described by (12) is well suited to this kind of architecture since neurons with different relaxation constants may be activated depending on the level of the output.

This behavior can be obtained by a network of neurons where both inhibitory and excitatory paths emanating from a common origin drive the same output class. An example of a four neuron-network as depicted in Figure 5. This network is asymptotically stable and it was trained to have the "peaked" response depicted in Figure 4. Similar structures exist in the cerebellum. The *granule-Purkinje* and *granule-Basket-Purkinje* paths (c.f. Figure 2.) are the excitatory/inhibitory paths emanating from the same common origin and affecting the same target output class.

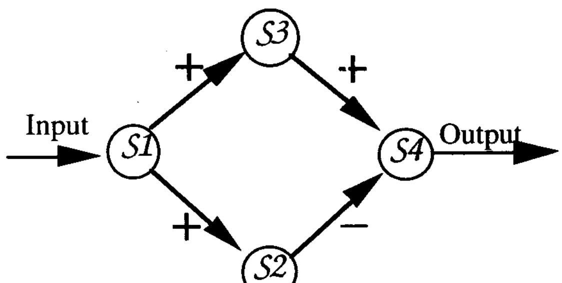

Fig. 5. A network of neurons exhibiting "peaked" response.

IV. APPLICATIONS

We have used the neural networks described above to successfully identify a number of nonlinear dynamical systems. These include a robot arm, the dynamics of boat under rudder input and the influence of the temperature on the magnitude of the pilot signal obtained from a network of high-frequency cable-television-distribution amplifiers. In the following sections, we shall present these identification experiments, together with convergence measurements.

A. *Identification of a PUMA 560 Robot*

A two-link robot arm is known to have its dynamic response governed by the differential equation

$$H(q)\ddot{q} + h(q,\dot{q})\dot{q} + F\dot{q} + g(q) = \Phi v \qquad (13)$$

where the state q contains the angle θ_1 that the first link makes with the vertical and the angle θ_2 that is formed between the two links; $H(q)$ is the 2x2 inertial matrix; $h(q,\dot{q})$ models the Coriolis and centripetal forces; F is the friction matrix; $g(q)$ represents the gravitational torque; and Φ is the voltage-to-torque conversion matrix [5]. All of these variables rely on many machine-specific factors, such as dimensions, weights, inertia, and joint friction. To obtain an accurate model, one measures directly as many variables as possible. This was done for a PUMA-560 robot. Lengths, masses, and inertias were obtained through direct measurement [2]. Variables which could not be easily directly measured were the matrices F and Φ representing four unknown scalars in total. Classical RLS parameter estimation was used to identify these variables, and the final response to the input vector shown in Figure 6 is shown in Figure 7. The deviation of the model from the actual response underlines the difficulty in identifying complex systems using traditional model based methods.

A neural network with an architecture as presented above was trained to identify the dynamic response for θ_1, the angle that the first link makes with the vertical. Both $v_1(t)$ and $v_2(t)$ (the actuator control voltages) were used as inputs to the system. The neural network had an architecture similar to that shown in Figure 3, except that $\mathcal{N}2$ was not included. Class $\mathcal{N}1$ contained 5 neurons as did the scheduler class.

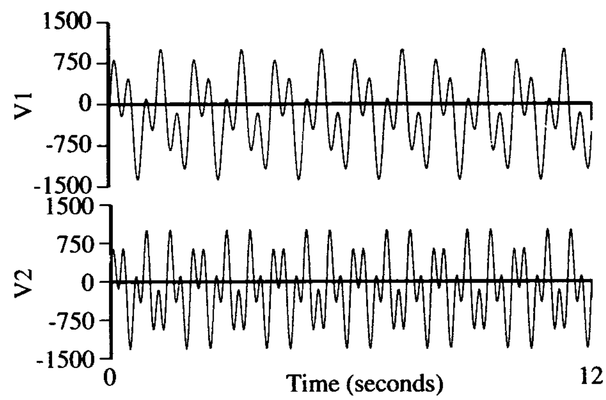

Fig. 6. The Control Voltages used to drive the robot.

Convergence of the response was rather slow; typically convergence was attained after several hundred thousands of training epochs. After training was completed, the neural network followed the actual response of the robot closely.The response is shown in Figure 8.

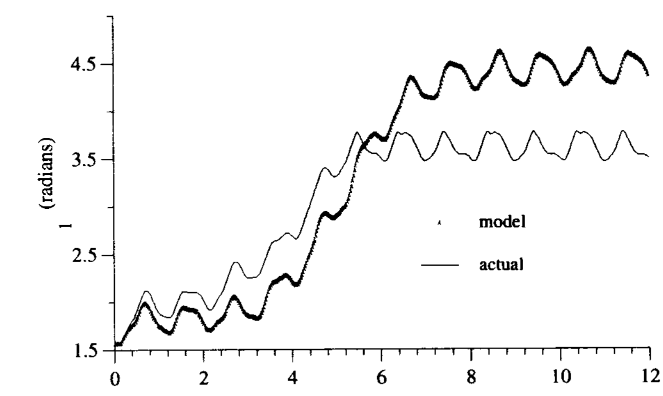

Fig. 7. Actual and model response to input shown in Figure 5.

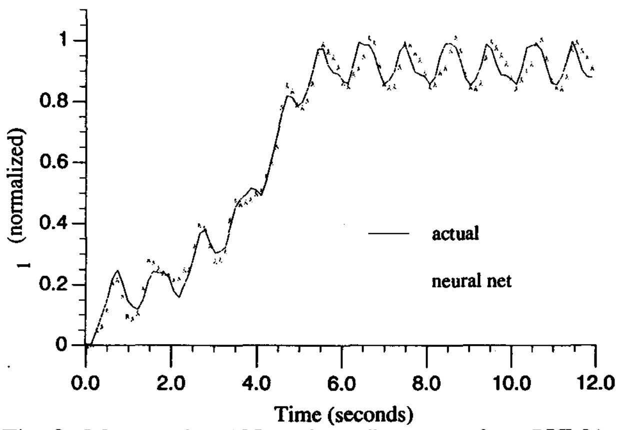

Fig. 8. Measured and Neural Net Response for a PUMA 560

The error as a function of the training epoch is depicted in Figure 9. As it can be seen, the error diminished very rapidly in the beginning, then the training spent quite a bit of time seeking isolated minima, but when it found them, the improvement was drastic and fast.

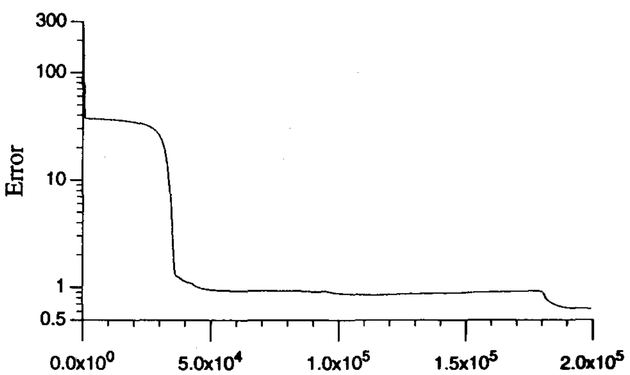

Fig. 9. The training error for the PUMA 560 experiment, as a function of the training epochs.

B. *Identification of a Boat*

A boat may be treated as a SISO system, with the rudder angle as the input and the heading as the output [1]. We have used a recurrent neural network to identify this behavior. Figure 10 shows the data which was used to train the neural network. This is equivalent to approximately 2 minutes of data collected from a boat and shows both the rudder angle and the heading of the boat.

The training method consisted of applying the input to the neural network, calculating its response, and using the measured heading to generate an error signal for weight adjustment. Figure 11 shows the response of the trained network to the training data. The vertical line delineates the data that were used to train the network, while the remaining of the figure depicts the response of the network to input that was not used to train the network.

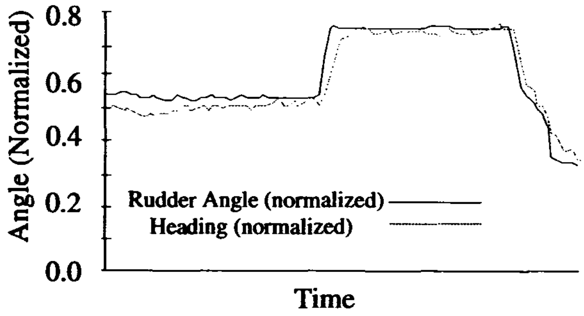

Fig. 10. Rudder Angle and heading during a Training Run

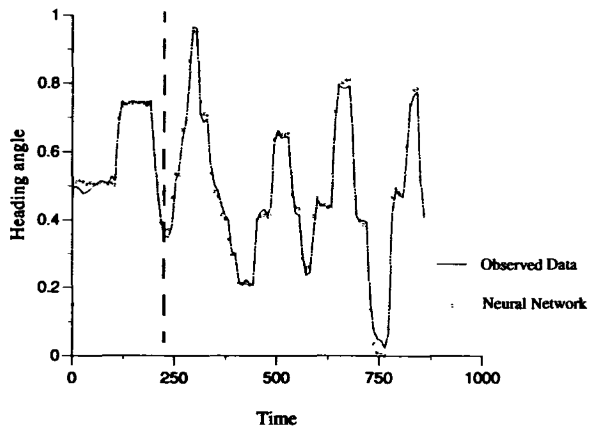

Fig. 11. Measured and Neural Net Response to 12 Minute Test Run

The neural network follows the measured response well. The difference between the measured and neural net response is due to the fact that the measurement noise in the heading is a stochastic process which cannot be predicted by the deterministic neural net model.

C. *Random techniques to speed the training convergence.*

Convergence is achieved rapidly once the training has reached the region of "attraction" of an optimum point. Finding the regions of "attraction" is considerably slow. We are using random perturbance during training to force the state to be dislodged from non interesting regions.

The first technique we use is that of exploratory searches. Several random disturbances to the weights are offered, and each is followed for a short number of epochs. The one that has reduced the error the most is chosen for a longer search. This technique can be applied at any point during the training, currently we have implemented it only at the start.The second technique, introduces smaller and smaller random disturbances to the weights as a function of the number of training epochs (akin to "simulated annealing").

Both techniques seem to speed the convergence significantly. An example is shown in Figure 12.

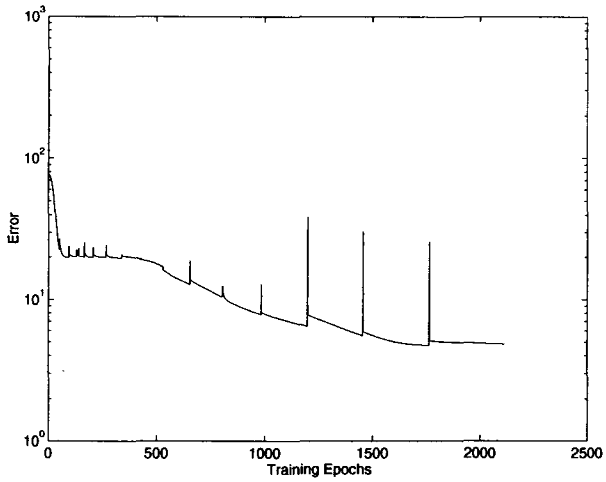

Fig. 12. The training error as a function of the training epochs for the network identifying the rudder angle. Random disturbances to the weights are introduced during the training. The sudden jumps in the error signify the introduction of a disturbance.

D. *Identification of the Pilot Measurements in a Cable-Television Network of High-Frequency Amplifiers*

A Cable Television Network incorporates a number of high frequency amplifiers forming (for conventional networks) a tree. In more advanced networks the structure incorporates a double ring from which subscriber drops emanate. In this work, we are focusing in conventionally structured tree networks. There are two categories of amplifiers, the ones belonging to the main trunk and the ones forming subscriber drops. Additionally, power supplies are located throughout the network, each one powering a limited number (typically three) of amplifiers. The majority of the main trunk amplifiers are equipped with a status monitor which uses a reverse channel to report the status of the amplifier to the head office. Subscriber drops and power supplies are not normally monitored.

The values of the monitored variables are allowed to vary within two intervals (warning and alarm) centered at nominal values. If a value is outside these predefined intervals then a warning or an alarm is issued.

There are several other modalities which manifest themselves as changes of behavior rather than significant changes in the measurements.

In order to detect the onset of such behavior changes and providing a diagnosis based on the properties of the ensuing behavior we have used recurrent neural networks identify the behavior, and we present examples of identification of pilot measurements.

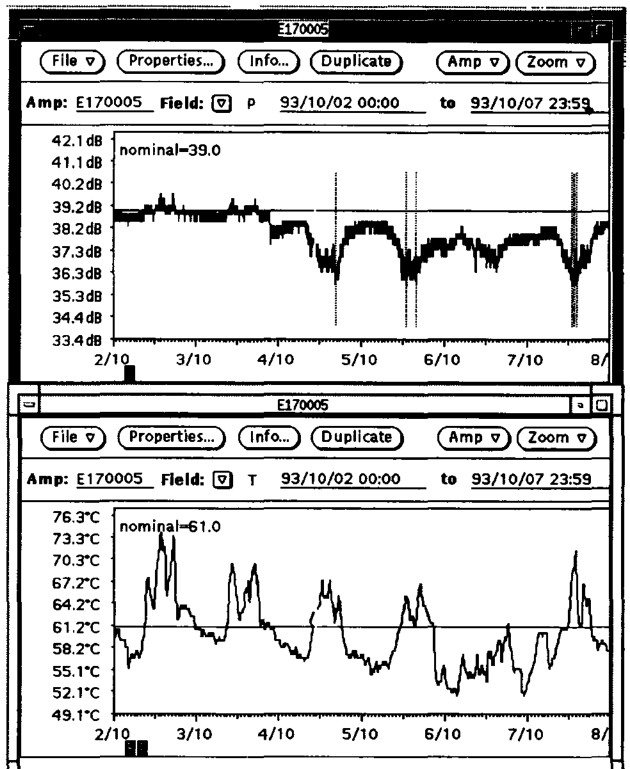

Fig. 13. Pilot and Temperature data over several days in October 1993. The shaded regions in the pilot data indicate alarms (i.e. measurements outside preset limits). Observe the fault initiation on October 3[d].

Although, the amplifiers are temperature compensated, it has been observed that all the measurements vary with the temperature. The typical variation is small when the amplifier is properly adjusted. Because of drifts or malfunctions, the amplifiers evolve to a "faulty" state where they exhibit an altered pattern of behavior. A typical example is presented in Figure 13. The behavior pattern depicted during the first two days (until October 3[d] at 21:40) is representative of a well tuned high-pilot amplifier. Observe the small, temperature correlated variations of the pilot level around the nominal value of 39 db. Suddenly, on October 3[d] at 21:40, a significant change in the pattern of behavior occurs. The standard fault detection techniques which are normally in use, alert the user only if the values of a measured parameter exceed some preset limits. In this case the pilot level did not exceed its threshold until the following day, some eighteen hours after the start of the new behavior in the evening of October 3[d], and then it stayed outside its nominal range for only a limited duration.

Such a belated reporting in conjunction with the short duration of the time during which a measurement stays outside its normal range, makes a diagnosis very difficult.

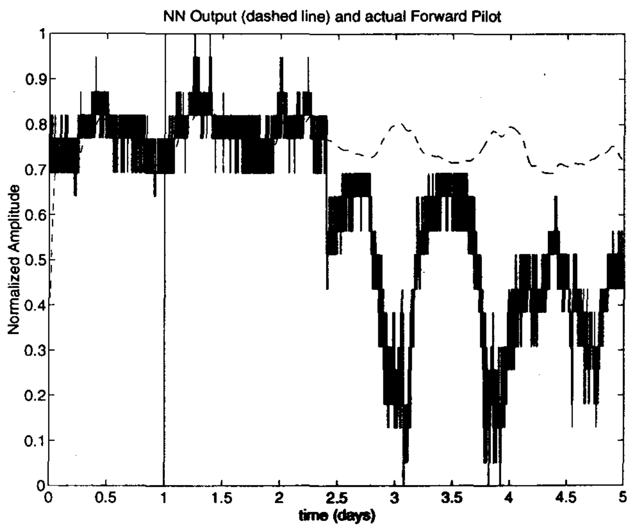

Fig. 14. Actual and Neural Network (dashed line) response for the pilot of E170005. The training set is delineated by the vertical line at day 1.

One may observe therefore that establishing a nominal range of values, is not the best way of detecting the onset of a "faulty" behavior pattern. In addition it does not accurately identify periods during which the behavior is "faulty", and thus it makes an accurate diagnosis of the fault problematic.

An accurate detection of a "fault" initialization and detection requires a model of the behavior of the measurement in time and its dependencies. Any deviation from the model, would denote the onset of a "fault", while the model of the "faulty" behavior pattern, if it could be established, would contribute to the diagnosis.

We have used recurrent neural networks, as presented in Section II above, to model the behavior of the pilot measurement and its dependence on the temperature of the enclosure. Figure 14. presents the response of the trained neural network (consisting of two classes each comprising 10 neurons, one being a scheduler class) together with the actual readings for the pilot.

The neural network was trained with data from October 1, delineated by the vertical line on day 1 in the plot. Observe the abrupt change of behavior at the fault.

V. Conclusions And Discussion

In this work we have presented a class of recurrent networks which are asymptotically stable. We have introduced a training methodology for networks belonging to this class, and used it to train networks that successfully identified a number nonlinear systems.

The systems which were used in our identification experiments were actual systems and included a robot, the dynamics of a boat and the dependence of the pilot-signal measurements on the ambient temperature of the enclosure of a high-frequency trunk amplifier.

The response of the trained networks follows closely that of the actual system, confirming the ability of these structures to accurately model the system dynamics.

Finally, we are currently using the model obtained for the dependence of the pilot-signal on the temperature to establish the onset of a "fault" by establishing the moment at which measurements start deviating from the values predicted by the model. The characteristics of the behavior pattern after the fault are indicative of the "fault" modality.

Acknowledgements

This work was supported through a grant by the Canadian Cable Labs Fund.

References

[1] A. Andekian, M.A.Sc. Thesis, University of Victoria, Victoria B.C., 1993.

[2] B. Armstrong, O.Khatib, and J. Burdick, "The Explicit Dynamic and Inertial Parameters of the PUMA-560 Arm", *IEEE Int'l Conference on Robotics and Automation*, 1986, pp. 510-518

[3] K.J. Astrom and B. Wittenmark, *Adaptive Control*, Addison-Wesley Publishing Company, 1989.

[4] N. J. Dimopoulos, A. Watkins, S. Neville, K. F. Li "An Expert Network Analyzer: Knowledge Acquisition, Fault Diagnosis and Prediction" *Proceedings 1993 DND Workshop on Advanced Technologies in Knowledge Based Systems and Robotics* Nov. 14-17 1993, Ottawa, Canada.

[5] M. Erlic, M.A.Sc. Thesis, University of Victoria, Victoria B.C., 1990.

[6] C. M. Jubien, N. J. Dimopoulos "Recurrent Neural Networks in Systems Identification" *Proceedings 1993 IEEE International Symposium on Circuits and Systems* pp. 2458-2461 (May 1993)

[7] C. M. Jubien, N. J. Dimopoulos "Identification of a PUMA-560 Two Link Robot Using a Stable Neural Network" *Proceedings 1993 International Conference on Neural Networks* (Apr. 1993).

[8] N. Dimopoulos, "A Study of the Asymptotic Behavior of Neural Networks," *IEEE Transactions on Circuits and Systems*, Vol. 36, No.5, pp. 687-694, May 1989.

[9] K.S. Narendra and K. Parthasarathy, "Identification and Control of Dynamical Systems Using Neural Networks," *IEEE Transactions on Neural Networks*, Vol. 1, No. 1, pp.4-27, March 1990.