975612

System Identification via Artificial Neural Networks:Applications to On-line Aircraft Parameter Estimation

S. Massoud Amin, Volker Gerhart, and Ervin Y. Rodin
Washington Univ.

1997 World Aviation Congress
October 13-16, 1997
Anaheim, CA

SAE The Engineering Society For Advancing Mobility Land Sea Air and Space® INTERNATIONAL

SAE International
400 Commonwealth Drive
Warrendale, PA 15096-0001 U.S.A.

AIAA

American Institute of Aeronautics and Astronautics
370 L'Enfant Promenade, S.W.
Washington, D.C. 20024

975612

System Identification via Artificial Neural Networks:Applications to On-line Aircraft Parameter Estimation*

S. Massoud Amin, Volker Gerhart, and Ervin Y. Rodin
Washington Univ.

ABSTRACT

In this report, the neural identification problem is outlined and the identifiability question for a general class of recurrent neural networks is addressed. As part of the intelligent flight control concept program, recurrent second-order neural networks are utilized in order to continuously identify critical stability and control parameters during flight. Our group at Washington University participated in Phase II, the online learning, with neural networks that learn new information during flight. In particular, a recurrent second-order neural network architecture with a robust filtered error learning algorithm was utilized to identify the dynamics of an F-15 aircraft.

While the emphasis of our work has been on the development and implementation of online neural network estimators, we shall also include results with and without the baseline network. Several examples including in-flight situations are presented and the effectiveness of the recurrent high-order neural networks is illustrated.

INTRODUCTION

The capability of artificial neural networks to model the behavior of large classes of uncertain nonlinear dynamical systems within a certain accuracy is a basic requirement in any application; such approximation guarantees are also necessary for the application of neural networks in identification and control of nonlinear dynamical systems. Artificial neural systems currently gain much insight from adaptive control theory. The recent developments in neuro-adaptive control motivated by results and tools from robust adaptive control theory have already enhanced the understanding of neural on-line system identification and control. Parameter estimation algorithms based on nonlinear optimization and stability theory for dynamical neural networks are presented. The theoretical difficulties in deriving stability and convergence results for well-known gradient-descent schemes in closed-loop real-time identification and control systems is pointed out. An improved on-line parameter estimation algorithm based on a Lyapunov-like approach with conditions for guaranteed stability is presented. The robustness of the algorithm in the presence of modeling errors is investigated.

The capability of neural network approximations of static and continuous functions is well known; approximation theorems with particular attention to the topology, structure, dimensionality and quality of the network have been addressed elsewhere in the literature. However, most theorems have provided little constructional information and serve as important existence results. With this as background, dynamical/recurrent neural networks are discussed as approximators of dynamical systems. Their application in dynamical system identification is quite intuitive and has proven to be very successful. Moreover, the increasing significance of nonlinear dynamical controller designs in nonlinear control systems raises the question of the application of various recurrent neural networks as dynamical controllers in nonlinear neuro-control systems. The design of dynamical neuro-controllers in nonlinear control with guaranteed performance has been and will continue to be of interest in the future.

Due to rising interest in dynamical neural network models within the control community, dynamical neural nets of various types have been considered. Several possibilities have been investigated, including a) static neural networks augmented with tapped delay lines which can be seen as intermediate recurrent network configurations; b) feedback connections of Hopfield-type networks; c) feedforward distributed dynamical neural networks, where each neuron has a local dynamical model governed by a first-order differential equation; and d) recurrent higher order neural networks where in addition to local dynamics of type (c), each unit can receive, as input, products of outputs from other units.

* This work was supported in part by a NASA-Ames through McDonnell Douglas Contract # MDA-Z50337, and by US AFOSR under Grant F49620-96-1-0151.

NEURAL APPROXIMATION

A general single input, single output (SISO) nonlinear system can be represented by the deterministic, discrete-time input/output description:

$$y(k+1) = f[y(k), y(k-1), \ldots, y(k-n+1); \\ u(k), u(k-1), \ldots, u(k-m+1)]. \quad (1)$$

This description is referred to as the nonlinear autoregressive moving-average model (NARMAX) [1]. According to [2]-[7] the NARMAX model is a natural representation for sampled nonlinear continuous-time systems. The input/output representation describes the input/output behavior of a system. Since the input/output behavior is all that can be seen by an external observer, its mathematical representation is of primary importance in systems theory. It is assumed that the function $f:\Re^{m+n} \rightarrow \Re$ is continuous, but not given explicitly. In fact, for many real sampled nonlinear systems their NARMAX models are very difficult to determine; this is in part due to the "curse of dimensionality" which would render a high-order function intractable when a large number of delay elements are required to accurately represent it. In general the nonlinear structure of f is unknown; therefore a means of approximating f is desired and necessary. Knowing a finite number of measurements, i.e. current and delayed inputs and outputs samples of a given physical system, one may pose the question of whether there exists a representation consisting of a finite number of known functions and real parameters that approximates the mapping f. Similarly one may seek an approximation of a continuous feedback control law. These questions point to the upcoming approach as relevant and important for both modeling and control. In fact, it will emerge that neural network models can be utilized as parametric representation structures for purposes of identification and control.

Notice however that the approximation capability of a given parametrized neural model does not imply the existence of a convergent identification algorithm for the parameters of this model, i.e. the convergence of an iterative learning algorithm for the construction of f. In fact, the approximation theory may guarantee the existence of a set of parameters for a neural model such that f can be approximated, whereas the construction of f (or better, the identifiability of this set of parameters) is a different issue which will be discussed in the next section. Thus approximation theory provides a pure existence result whereas identification mainly deals with a construction procedure. Nevertheless, before considering the identification problem, i.e. how to find the appropriate parameter values for a given set of data, the fundamental representation problem must be addressed. Answers must be given to the basic questions: which class of mappings f can be approximated by which neural models, and how well?

NEURAL ARCHITECTURES

Neural networks are deterministic nonlinear systems described by algebraic expressions or ordinary differential equations. This section presents continuous-time state-space representations of recurrent neural networks. The most significant advances in the development of a systematic body of transparent and constructive design principles have been made in neural adaptive control systems. These were reviewed in [8]. Although major results in approximation and identification of systems using neural networks are available, a small yet growing minority in the control community is familiar with them. Due to the back-propagation learning algorithm, static multi-layer feedforward neural networks, also known as multi-layer perceptrons, have become very popular and utilized as approximators of static nonlinear mappings in many different applications[†]. In order to deal with time-varying mappings such as input/output mappings of a dynamical system, static neural networks were augmented with so called tapped delay lines at their inputs which allow them to access a constant time-frame of the mapping at each time step, thus capturing the model given in EQ (1). With the incorporation of feedback connections and delay elements between units, static neural networks were turned into recurrent architectures. Recurrent neural networks are characterized by their internal memory and thus are very suitable for imitating the behavior of dynamical systems.

In addition to feedback between units several approaches have distributed dynamical elements throughout the network; such networks have recently been rediscovered as dynamical network models in the context of identification and control of nonlinear dynamical systems.

Recurrent Neural Networks

While feedforward networks are static mappings between two information domains, the structure of recurrent neural networks incorporates dynamical behavior through feedback connections. In practice recurrent neural networks have been successfully applied in real-time recognition of temporal patterns and identification of dynamical systems (see for example [19-37]). These kind of tasks can hardly be solved by memoryless, static networks. Figure 1 depicts a recurrent neural network; such a network may still have a layered structure where, in addition to feedforward connections, unit outputs are fed back as inputs to units in previous layers. In a fully recurrent network the layered structure is lost as seen in the figure. Each unit receives inputs from all other units in the network including feedback signals of its own output. The M input units get external inputs from outside the network. The outputs of the P output units are the output of the overall system. Notice the special case when a input unit is also an output unit in the same setup. The remaining $N^* = N - P - M$ units represent the network's internal hyperstate. The state-space description of the dynamical unit (in a Hopfield network) is given by:

$$T_i \dot{x}_i = -x_i + \sum_{j=1}^{N} w_{ij} y_j + u_i, \quad x(0) = x_0, \\ y_i = h(x_i), \quad (10)$$

[†] The interested reader may refer to [9]-[18] for introductory textbooks and collections of papers on neural networks.

where u_i is an external input to the unit. Obviously u_i =0 for all non-input units. The overall dynamics of this network are described by:

$$\begin{aligned} T\dot{x} &= -x + W\bar{y} + u, \quad x(0) = x_0, \\ \bar{y} &= H(x), \\ y &= C\bar{y}, \end{aligned} \tag{11}$$

where T, W, x, $\bar{y}$, y and u are appropriate matrices and vectors respectively and H is the known operator that applies the nonlinear function h to each of the elements of the vector x. Notice that for a given H the triple of matrices (W,T,C) uniquely determines the network dynamics. The state x_i in EQ (10) can be interpreted as the potential or short-term memory activity of a biological neuron.

The term $-T_i^{-1}x_i$ describes the passive decay of activity at rate $-T_i^{-1}$. Indeed, from a biological standpoint the choice of dynamical units has been affirmed in the literature as very plausible. First-order recurrent neural network architectures in the form of EQ (10) were first studied by Hopfield; important results concerning the stability of such networks have been accomplished [38], [39] and [41]. In proving the convergence to stable states for a symmetric weight matrix using Lyapunov's direct method, Hopfield pointed out their applicability as content-addressable memory (CAM). Later on, in [41], the absolute stability of global pattern formations was proven for a more general model with symmetric weights including the Hopfield neurons.

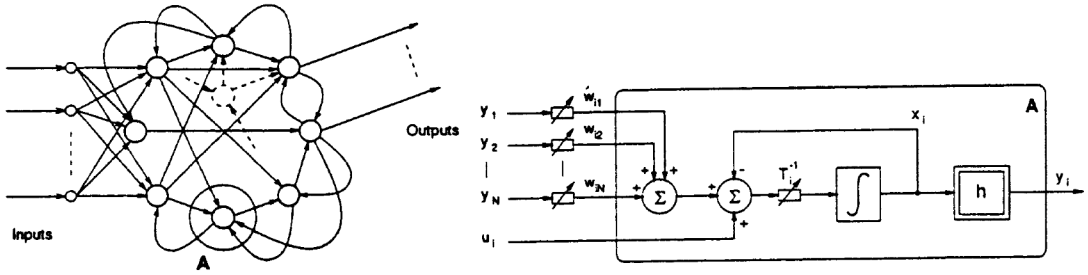

Figure 1: Recurrent neural network

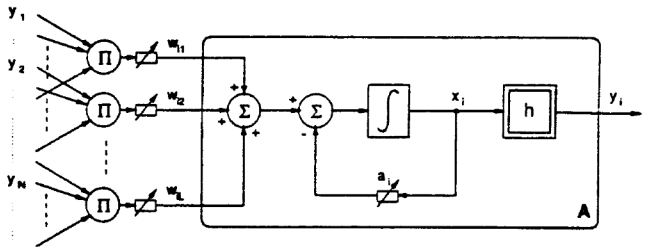

Figure 2: Recurrent higher-order neural network

Recurrent Higher-Order Neural Networks (RHONN)

The structure of a dynamical unit of a recurrent higher-order neural network is depicted in Figure 2. The dynamical components are distributed throughout the network similar to recurrent first-order neural networks in Figure 1. The network has higher-order interactions between neurons,where the input to a unit is not only a linear combination of the components of outputs of previous units but also of their products. Higher-order neural networks have a superior storage capacity, which increases with the number of connections between units [42], [43].

Stability properties for fixed weight values are studied in [44]. Recurrent higher-order networks have recently been utilized for identification and control of dynamical systems in [45], [28-37]. A general higher order dynamical unit of a network with memoryless, linear inputs and N dynamical units can be described by:

$$\dot{x}_i = -a_i x_i + \sum_{k=1}^{L} w_{ik} \prod_{j \in I_k} y_j^{d_j}, \tag{12}$$

where $\{I_1,\ldots,I_L\}$ is a collection of L not-ordered subsets of $\{1,2,\ldots,M+N\}$, a_i is a real coefficient, w_{ik} are the weight

parameters of the neural network and d_{jk} are nonnegative integers. The order of the network is determined by

$$\max_k \sum_{j \in I_k} d_{jk}.$$

Notice that $y_i = u_i$ for $i = 1,\ldots,M$ where u_i is an external input to the overall network, such that

$$\begin{aligned} \bar{y} &= \left[y_1,\ldots,y_M,y_{M+1},\ldots,y_{M+N}\right]^T \\ &= \left[u_1,\ldots,u_M,h(x_1),\ldots,h(x_N)\right]^T, \end{aligned} \tag{13}$$

where h are nonlinear continuous functions defined earlier. Introducing the parameter vector:

$$b_i^T = \left[w_{i1},\ldots,w_{iL}\right] \tag{14}$$

and the input vector of one unit:

$$g = \left[\prod_{j \in I_1} y_j^{d_{j1}},\ldots,\prod_{j \in I_L} y_j^{d_{jL}}\right]^T, \tag{15}$$

where $b_i, g \in \Re^L$, the mathematical description of the local unit dynamics can be restated for $i=1,\ldots,N$ as:

$$\dot{x}_i = -a_i x_i + b_i^T g \tag{16}$$

then introduce the network state vector $x \in \Re^N$

$$x = \left[x_1,\ldots,x_N\right]^T, \tag{17}$$

the system matrix $A \in \Re^{N \times N}$

$$A = -diag\left\{a_1,\ldots,a_N\right\}, \tag{18}$$

the weight parameter matrix $B \in \Re^{N \times L}$

$$B = \left[b_1,\ldots,b_N\right]^T. \tag{19}$$

Notice that g in EQ (15) depends on the external network inputs and the network states, i.e. $g = g(x,u)$. Then the dynamical behavior of the overall system can be described by:

$$\begin{aligned} &\dot{x} = Ax + Bg(x,u) \quad x(0) = x_0, \\ &y = C\bar{y}. \end{aligned} \tag{20}$$

Note that the general representation in EQ (20) includes the one of EQ (11) as a special case.

Approximate Realization of Dynamical Systems

We began with the question of whether a representation exists consisting of a finite number of known functions and real parameters that approximates a continuous mapping f of a dynamical system (1). Replacing the function f in EQ (1) with a feedforward neural network, it is clear that the resulting model can approximate the dynamical system. Moreover, introducing a multiple input, multiple (MIMO) system with a number of q inputs and p outputs, the $f: \kappa \rightarrow \Re^p$ where κ is a compact subset of $\Re^{qm+pn}$ can be approximately realized by a feedforward neural network. This application will be discussed further in this subsection.

In the past, classical methods for identifying nonlinear systems have often used static polynomial or sinusoidal nonlinearities with linear dynamical blocks, for example Hammerstein [46] or Wiener Operators and Volterra functional series [3]. These provide an adequate representation of a wide class of nonlinear systems, but often require several hundred parameters to characterize even simple systems. The excessive computational effort required to estimate the unknown parameters, the difficulty of interpreting the results and the necessity of special input signals are further disadvantages of these functional series methods. The usefulness of these system descriptions for identification and control purposes is therefore limited and alternative representations are required. Neural networks provide a promising approach, whose fast parallel computation using inexpensive hardware is hoped to make a difference in the future. They have already been successfully applied by the control community, although a rigorous system theoretic discussion of analysis and design is still in its very early stages. In order to reconstruct dynamical behavior it is necessary to store past information about system inputs, states and outputs in the network. A network model must have a memory. This can be realized in the form of delay elements. For the first time, a systematic approach for the integration of feedforward neural networks into traditional model reference adaptive control of dynamical systems was presented in [22], which initiated both a new wave of neural network applications in control systems and, more importantly, a theoretical discussion that intended to develop a consistent engineering methodology for the neuro-adaptive control of nonlinear dynamical systems recently. Narendra introduced the concept of a *tapped delay line* that incorporates delay elements at the inputs of a static feedforward neural network. The approach is shown in Figure 3. While in the depicted setup of a *series-parallel identification model* the actual plant outputs are delayed, the *parallel identification model* uses the fed back outputs of the neural network as input as seen in EQ (1).

The appropriate design of tapped delay lines for the approximation of the dynamics of a given plant demands *a priori* knowledge of the plant or some physical insight (such as the order and relative degree). Problems well known from traditional system identification and adaptive control (see for example [46-47]) have been encountered. Some knowledge of the physics of the plant should make the designer of the identification scheme able to choose a sufficient length for the tapped delay lines such that an approximation of the plant is possible.

First consider the series-parallel identification setup in Figure 3. In [7] it is proved that assuming that the given plant is of approximately finite memory, there exists a model of a two-layer feedforward neural network with tapped delay lines that approximates the plant up to any degree of accuracy. The approximation of dynamical systems is considered as approximations of continuous functionals on a compact set. Comparing EQ (1) with the setup in Figure 3 it is clear that a two-layer feedforward neural network with delay elements can approximate a dynamical system of finite memory. Now consider a parallel identification approach, i.e. the network outputs are fed back over time delays. If the identification is

off-line then approximation errors will be fed back and will eventually add up to an accumulated error over time.

Remark 1: Notice that the model of a feedforward neural network augmented with delays represents a simple recurrent network. For the continuity of the discussion it is presented here. In the next section approximation properties of more general recurrent networks shall be treated.

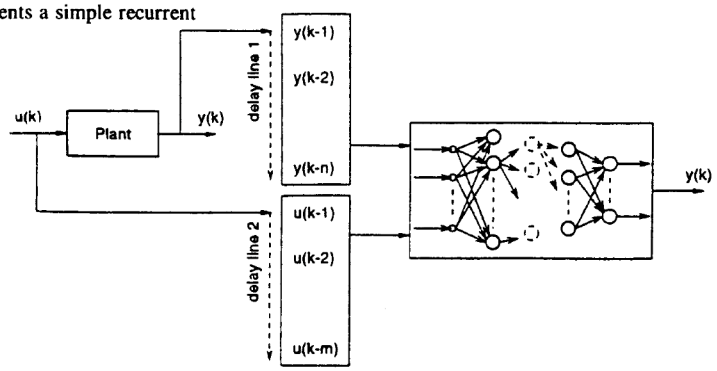

Figure 3: Tapped delay line setup

Consider a general autonomous continuous-time nonlinear dynamical system:

$$\begin{aligned} \dot{x}(t) &= f(x(t),u(t)), \quad x(0) = x_0, \\ y(t) &= h(x(t),u(t)), \end{aligned} \tag{21}$$

where $x(t) \in \mathrm{X} \subset \Re^n$ is the state, $u(t) \in U \subset \Re^m$ is the input, $y(t) \in \Re^p$ is the output vector, $f : \Re^n \to \Re^n$ is a smooth mapping and $t \in \Re^+$ is the temporal variable. Introduce the new state vector $\bar{x} = [x,u,y]^T$ such that

$$\begin{aligned} \dot{u} &= v, \\ \dot{y} &= \frac{\partial h(x,u)}{\partial x} f(x,u) + \frac{\partial h(x,u)}{\partial u} v, \end{aligned} \tag{22}$$

where v is the new input vector[‡]. The augmented system results in

$$\begin{aligned} \dot{\bar{x}} &= f(\bar{x},\bar{u}), \quad \bar{x}(0) = \bar{x}_0, \\ y &= C\bar{x}, \end{aligned} \tag{23}$$

where C is a constant matrix extracting the vector y from x. Thus without loss of generality it will be assumed that the system output is linear in the states[§]. Furthermore assuming that (21) is completely observable, for simplicity let $C = I_n$, where I_n is the $n \times n$ identity matrix. Hence, neglect the output equation and consider the system dynamics be given as the first-order state equation:

$$\dot{x} = f(x,u). \tag{24}$$

Before proving Theorem 1, a bit of background is needed: Funahashi proved a theorem which guarantees the approximate realization of continuous mappings by two-layer networks whose output functions for the hidden layer are sigmoidal, and whose neuron activation functions for input and output are linear in the sense of uniform topology [48].

Theorem (Funahashi): Let $\Phi(x)$ be a non-constant, bounded and monotone increasing continuous function. Let κ be a compact subset of $\Re^n$ and fix an integer $k \geq 2$. Then any continuous map $f : \kappa \to \Re^m$ defined by $f(x) = (f_1(x),\ldots,f_m(x))$ can be approximated in the sense of uniform topology on κ by k-layer feedforward neural networks whose output functions for hidden layers are $\Phi(x)$, and whose output functions for input and output layers are linear.

In other words, the above Theorem states that for any continuous map $f : \kappa \to \Re^m$ and an arbitrary $\varepsilon > 0$, there exists a k-layer network whose input/output mapping is given by $\hat{f} : \kappa \to \Re^m$ such that $\max_{x \in \kappa} d(f(x),\hat{f}(x)) < \varepsilon$, where $d(.)$ is a metric which induces the usual topology of $\Re^m$. The class of functions for $\Phi(x)$ specified by the above Theorem include sigmoidal functions. Hornik proved an analogous result on a more abstract level by using the Stone-Weierstrass Theorem [49]. Stinchcombe and White specify a more general class of functions $\Phi(x)$. They prove that all functions whose mean value is different from zero and whose L_p-norm is finite for $1 \leq p < \infty$ are applicable [50]. Now we shall state and prove the result:

[‡] For simplicity the dependence on time will not be written explicitly.

[§] Notice that dynamics in Eqn (23) are affine, i.e. $\dot{\bar{x}} = \tilde{f}(\bar{x}) + \sum_{i=1}^{m} g_i(\bar{x}) v_i$ with $g_i(\bar{x}) \in \Re^n$, for $i = 1,\ldots,m$.

Theorem 1 (Parallel Identification Model Approximation): Let D be an open subset of $\Re^{n+m}$ and $f: D \to \Re^n$ be a continuously differentiable mapping. Suppose that (24) defines a dynamical system on D. Let K be a compact subset of D and consider trajectories of the system on the interval $[0,T]$ with $0 < T < \infty$. Let $\hat{f}$ denote a two-layer feedforward neural network of the form $\hat{f} = C\,\sigma(B[x,u]^T)$. Then for an arbitrary $\varepsilon > 0$ there exist weight parameter matrices B^* and C^*, such that if the smooth mapping f is replaced by $\hat{f}$ then for any trajectory $\{x(t); 0 \le t \le T\}$, the approximated state vector $\hat{x}$ satisfies

$$\max_{t\in[0,T]} \|x(t) - \hat{x}(t)\| < \varepsilon, \qquad (25)$$

where it is assumed that $\hat{x}(0) = x(0) \in K$.

Proof: Let

$$K_\varepsilon = \left\{(\hat{x},\hat{u}) \in \Re^{n+m} \,\middle|\, \|(\hat{x},\hat{u}) - (x,u)\| \le \varepsilon, (x,u) \in K \right\} \qquad (26)$$

Notice that K_ε itself is a compact subset of $\Re^{n+m}$ and $K \subset K_\varepsilon$ where ε is the required degree of approximation. By Funahashi's Theorem it is possible to approximate any continuous mapping f by a two-layer sigmoidal feedforward neural network $\hat{f}$ on the compact set K_ε, i.e. for any $\varepsilon_1 > 0$ there exist weight matrices of appropriate dimension B^*, C^* in

$$\hat{f}(x,u) = C^*\,\sigma(B^*[x,u]^T), \qquad (27)$$

such that

$$\max_{(x,u)\in K_\varepsilon} \|f(x,u) - \hat{f}(x,u)\| < \varepsilon_1. \qquad (28)$$

Notice that the bias vector w is incorporated with the input vector u. Since the state vector is fed back, the approximated state dynamics are given by

$$\dot{\hat{x}} = C^*\,\sigma(B^*[\hat{x},u]^T) \qquad (29)$$

where $\hat{x}$ is the approximated state vector. From the latter and (24) the state error between approximation model and the ideal system $\tilde{x} = \hat{x} - x$ is described by the error dynamics

$$\begin{aligned}\dot{\tilde{x}} &= C^*\sigma(B^*[\hat{x},u]^T) - f(x,u)\\ &= C^*\sigma(B^*[\hat{x},u]^T) - C^*\sigma(B^*[x,u]^T) + \\ &\quad C^*\sigma(B^*[x,u]^T) - f(x,u).\end{aligned} \qquad (30)$$

Integrating the latter and taking the norm results in

$$\begin{aligned}\|\tilde{x}(t)\| \le \int_0^t \|&C^*\sigma(B^*[\hat{x}(\tau),u(\tau)]^T) - C^*\sigma(B^*[x(\tau),u(\tau)]^T)\|\,d\tau\\ &+ \int_0^t \|C^*\sigma(B^*[x(\tau),u(\tau)]^T) - f(x(\tau),u(\tau))\|\,d\tau.\end{aligned} \qquad (31)$$

Since $\hat{f}(x,u)$ is a continuously differentiable function, it is Lipschitz-continuous (locally Lipschitzian) on the compact domain K_ε, i.e. there exist a finite constant k_T such that for any (x_1,u_1), $(x_2,u_2) \in K_\varepsilon$

$$\|\hat{f}(x_1,u_1) - \hat{f}(x_2,u_2)\| \le k_T\left(\|x_1 - x_2\| + \|u_1 - u_2\|\right), \qquad (32)$$

where $k_T = k_T(B^*,C^*)$. The Lipschitz condition is fundamental for the existence and uniqueness of a solution of an ordinary differential equation. It guarantees a unique solution of the approximation model.

Assuming that $(x,u) \in K_\varepsilon$ for all $t \in [0,T]$ and using the approximation property (4.31), it follows

$$\begin{aligned}\|\tilde{x}(t)\| &\le \int_0^t \|\hat{f}(\hat{x}(\tau),u(\tau)) - \hat{f}(x(\tau),u(\tau))\|\,d\tau + \varepsilon_1 t\\ &\le \int k_T \|\tilde{x}(\tau)\|\,d\tau + \varepsilon_1 T.\end{aligned} \qquad (33)$$

Finally apply the Bellmann-Gronwall Lemma to obtain

$$\|\tilde{x}(t)\| \le \varepsilon_1 T e^{k_T T}. \qquad (34)$$

Choosing ε_1 in the above fundamental approximation theorem as

$$\varepsilon_1 = \frac{\varepsilon \cdot e^{-k_T T}}{2T} \qquad (35)$$

then

$$\|\tilde{x}(t)\| \le \frac{\varepsilon}{2}. \qquad (36)$$

In order to verify that the result in (36) indeed holds for all $t \in [0,T]$, assume that (x,u) leaves the domain K_ε at time $t_1 < T$, such that $(x(t_1),u(t_1)) \in \partial K_\varepsilon$ is on the boundary of K_ε. By carrying out the same analysis as above for $t \in [0,t_1]$, we note that the result (36) contradicts the assumption that (x,u) leaves the domain K_ε at time $t_1 < T$. Therefore (36) is true for all $t \in [0,T]$. The error between the neural model and the physical system depends on the approximation of the continuous mapping f and the time T. ε_1 decreases exponentially with the size T of the considered time interval. This puts a very strong approximation demand on the feedforward network. Clearly, the better the approximation of the mapping f is, the better will be the approximation of the dynamics (24) for a greater T[**].

Remark 2: The approximation ability in the series-parallel identification case can be seen from EQ (30). Since actual plant values are fed into the neural network model the error dynamics are:

[**] Several proofs similar to the above have been used in other problems that integrate neural networks as nonlinear functions approximators (e.g., [45] and [52]). They rely on standard results from the theory of ordinary differential equations, e.g. [53] or [54].

$$\left\|\dot{\bar{x}}(t)\right\| = \left\|C^*\sigma(B^*[x,u]^T) - f(x,u)\right\| < \varepsilon_1 \quad (37)$$

It follows that

$$\left\|\bar{x}(t)\right\| < \varepsilon_1 T. \quad (38)$$

Remark 3: For a recent discussion on the reconstruction of nonlinear systems using tapped delay lines see [51].

Remark 4: Recall the general autonomous continuous-time nonlinear dynamical system (21); it can easily be converted into affine form by passing the input through integrators. Let the new state vector be $\bar{x} = [x,u]^T$. Introduce the new input vector v such that $\dot{u} = v$. Then the affine system dynamics are given by

$$\begin{bmatrix}\dot{x}\\ \dot{u}\end{bmatrix} = \begin{bmatrix}f(x,u)\\ 0\end{bmatrix} + \begin{bmatrix}0\\ I\end{bmatrix} v. \quad (39)$$

Considering the class of affine nonlinear systems described by

$$\begin{aligned}\dot{x}(t) &= f(x) + \sum_{i=1}^{m} g_i(x)u_i, \quad x(0) = x_0,\\ y(t) &= h(x),\end{aligned} \quad (40)$$

where $g_i(x) \in \Re^n$ for $i = 1,\ldots,m$ and affine denotes that the control input u appears linearly with respect to the matrix $G = [g_1,\ldots,g_m]$, one can construct a recurrent neural network model by replacing the nonlinear mappings f, G and h by feedforward neural networks.

General Recurrent Neural Networks

In the following it will be shown the internal state of the output units of continuous-time recurrent neural networks can approximate a given dynamical system to any precision. One can indeed embed an N-dimensional dynamical system into a higher dimensional system which defines a recurrent neural network. While [55] proves the approximation theorem for a trajectory of a discrete dynamical system by the use of the fundamental approximation theorem, the corresponding continuous-time study can be found in [56]. Recall the recurrent neural network model given by EQ(11)

$$\begin{aligned}T\dot{x} &= -x + W\bar{y} + u, \quad x(0) = x_0,\\ \bar{y} &= H(x),\\ y &= C\bar{y},\end{aligned}$$

The existence and uniqueness of a solution of the differential equation (11) is guaranteed by the next theorem.

Theorem 2 (Existence and Uniqueness): Let D be an open subset of $\Re^n$. Let $u(t) \in U \subset \Re^m$ be a bounded input and $H: D \to \Re^n$ be a bounded and continuously differentiable mapping. Let $W, T \in \Re^{n\times n}$ be constant matrices with $T_{ii} > 0$ and $T_{ij} = 0$ for $i \neq j$. Then for $t \in [0,\infty)$ there exists a unique solution of the differential equation:

$$\begin{aligned}T\dot{x} &= -x + Wy + u, \quad x(0) = x_0,\\ y &= H(x).\end{aligned} \quad (41)$$

Proof: By the assumption that u is bounded and $H(x)$ is a bounded mapping, there exist constants $M_1, M_2 > 0$ such that $|u_i| < M_1$ and $|H_i(x)| < M_1$ for $i = 1,\ldots,n$. Therefore for $F(x,u) = W \cdot H(x) + u$ there exists a constant $M > 0$ such that $|F_i| < M$ for $i = 1,\ldots,n$. Now consider the solutions of the differential equations

$$\begin{aligned}T_{ii}\dot{x}_i &= -x_i - M,\\ T_{ii}\dot{x}_i &= -x_i + M,\end{aligned} \quad (42)$$

which are

$$x_i(t) = x_i(0)\,e^{-\frac{t}{T_{ii}}} \pm M \cdot (e^{-\frac{t}{T_{ii}}} - 1) \quad (43)$$

Thus

$$|x_i(t)| \le |x_i(0)| + M \equiv N_i. \quad (44)$$

Let $N = \max_i N_i$. Then

$$\left\|x(t)\right\| < \sqrt{n}\,N \quad (45)$$

i.e. x is bounded on the existing interval of solution. Since the right side of (41) is continuously differentiable in x, it Lipschitz-continuous on D. Therefore there exits a unique solution x over $[0,\delta]$ provided the number δ is sufficiently small. But since x is bounded, as seen above, there exists no finite escape time and thus the solution must exist uniquely over $[0,\infty)$.

Remark 5: "At a first glance the condition of Lipschitz-continuity appears to be extremely restrictive, since it is known that "almost all" continuous functions are not differentiable and thus not Lipschitz-continuous. Nevertheless, it can be shown that differential equations with unique solutions are prevalent in the sense "almost all" differential equations with continuous functions f have unique solutions." [57]

Funahashi [56] proves two theorems for recurrent networks of the form (11) with external input $u=0$. While the first theorem proves the approximation capability of a autonomous dynamical system, the second theorem guarantees the approximation of continuous mappings. These results are stated next.

Theorem 3 (Funahashi) : Let D be an open subset of $\Re^n$ and $f: D \to \Re^n$ be a continuously differentiable mapping. Suppose that $\dot{x} = f(x)$ defines a dynamical system on D. Let K be a compact subset of D and consider trajectories of the system on the interval $[0,T]$ with $0 < T < \infty$. Then for an arbitrary $\varepsilon > 0$ there exists an integer N and a recurrent neural network with n output units and N hidden units (hyperstates) such that for any trajectory

$\{x(t); 0 \le t \le T\}$ of the system with initial value $x(0)$ in K and an appropriate initial state of the network,

$$\max_{t\in[0,T]} \|x(t) - \hat{x}(t)\| < \varepsilon \quad (46)$$

holds, where $\hat{x}$ is the internal state of output units of the network.

Theorem 4 (Continuous Mapping Approximation): Let $f:[0,T] \to \Re^n$ be a continuous function where $0 < T < \infty$. Then for an arbitrary $\varepsilon > 0$ there exists an integer N and a recurrent neural network with n output units and N hidden units (hyperstates) such that

$$\max_{t\in[0,T]} \|f(t) - \hat{x}(t)\| < \varepsilon \quad (47)$$

where $\hat{x}$ is the internal state of output units of the network.

Recently recurrent higher order neural network of the form (20) have been discussed in the context of system identification [45]. The following theorem guarantees the existence of an approximation of the form (20) for any dynamical system.

Theorem 5 (Recurrent Higher-Order Neural Networks Approximation): Suppose the nonlinear system dynamics (24) and the recurrent higher order neural network (20) are initially in the same state $x(0) = \hat{x}(0)$ where $\hat{x}$ is the state of the network model. Then for any $\varepsilon > 0$ and any *finite* $T>0$, there exists an integer L and a matrix $B^* \in \Re^{n\times L}$ such that

$$\sup_{t\in[0,T]} \|\hat{x}(t) - x(t)\| < \varepsilon \quad (48)$$

Proof: Since the arguments are similar to the proof of Theorem 1 only a sketch of the proof will be outlined here. Let the sets D, K and K_ε be given as before. A tricky step is to restate EQ (24) as

$$\dot{x} = Ax + g(x,u), \quad (49)$$

by introducing $g(x,u) = f(x,u) - Ax$, where A is the stable system matrix of the neural model (20). Then the error dynamics are given by

$$\dot{\tilde{x}} = A\tilde{x} + g(x,u) - B\hat{g}(\hat{x},u), \quad (50)$$

where $\hat{g}$ belongs to the neural model. Integrate the error dynamics and take the norm. By applying the triangle inequality and Cauchy-Schwarz inequality we get:

$$\|\tilde{x}(t)\| \le \int_0^t \|e^{A(t-\tau)}\| \cdot \|g(x(\tau),u(\tau)) - B\hat{g}(x(\tau),u(\tau))\| d\tau + \int_0^t \|e^{A(t-\tau)}\| \cdot \|B\| \cdot \left\| \begin{matrix} \hat{g}(x(\tau),u(\tau)) \\ -\hat{g}(\hat{x}(\tau),u(\tau)) \end{matrix} \right\| d\tau \quad (51)$$

where the norm $\|\cdot\|$ for matrices denotes the Frobenius matrix norm. By the Stone-Weierstrass Theorem it can be shown that for any $\varepsilon_1 > 0$ there exits a weight parameter matrix B^* such that

$$\max_{(x,u)\in K_\varepsilon} \|g(x,u) - B^* \hat{g}(x,u)\| < \varepsilon_1. \quad (52)$$

Recall the well known result for a Hurwitz matrix A that there exist scalars $\alpha, \beta > 0$ such that

$$\|e^{At}\| \le \beta e^{-\alpha t}. \quad (53)$$

Under the further assumption that $\hat{g}(x,u)$ is Lipschitz-continuously differentiable, it is Lipschitz-continuous with the Lipschitz constant k_T. Assuming that $(x,u) \in k_\varepsilon$ for all $t \in [0,T]$ it results in

$$\|\tilde{x}(t)\| \le \varepsilon_1 \beta \int_0^t e^{A(t-\tau)} d\tau + \beta k_T + \|B^*\| \int_0^t \|\tilde{x}(\tau)\| \cdot e^{-\alpha(t-\tau)} d\tau. \quad (54)$$

Now use the Bellmann-Gronwall lemma and assuming that the given α, β in (53) satisfy $\alpha < \beta k_T \|B^*\|$. If this is not true, i.e. $\alpha > \beta k_T \|B^*\|$, simply choose a smaller $\bar{\alpha}$ that satisfies the condition. Since $\bar{\alpha} < \alpha$, (53) remains true for $\bar{\alpha}$. Then

$$\|\tilde{x}(t)\| \le \frac{\varepsilon_1 \beta}{\beta k_T \|B^*\| - \alpha} \left[e^{(\beta k_T \|B^*\| - \alpha)T} - 1 \right] \quad (55)$$

and choosing ε_1 appropriately one can get $\|\tilde{x}(t)\| \le \frac{\varepsilon}{2}$. The final contrapositive argument to ensure that this indeed holds for all $t \in [0,T]$ is exactly the same as in the proof of Theorem 1.

Remark 6: Notice that in EQ (49) the mapping $g(x,u)$ and a corresponding nonlinear output function $h(x,u)$ can be replaced by a feedforward network in order to construct an overall recurrent neural model. The approximation property of a general nonlinear system of the form (21) using the so constructed network is proven in [52].

Remark 7: Constructing a type of recurrent network configuration by augmenting the static network architecture with dynamical elements in the form of stable filters is very common. In fact, as already mentioned for affine systems, one often refers to known and well analyzed classes of nonlinear systems and replaces their unknown nonlinearities by neural networks. Thus the new system is parameterized by neural network models with known underlying structure but unknown parameters. This approach incorporates certain neural network abilities and often leads to a better understanding of the overall neural system model by applying classical nonlinear analysis methodologies.

Remark 8: A model represents a way of predicting future outputs of the physical system. Therefore a recurrent neural model

$$\dot{\hat{x}} = \hat{f}(\hat{x}, u, \theta_1), \quad \hat{x}(0) = x_0,$$
$$\hat{y} = \hat{h}(\hat{x}, u, \theta_2),$$

where θ_1 and θ_2 are adjustable parameter vectors, is a one-step-ahead predictor for the underlying system description

$$\dot{x} = f(x,u), \quad x(0) = x_0,$$
$$y = h(x(,u).$$

Remark 9: The problem of approximating a nonlinear dynamical system by a recurrent neural network has been recently formulated as the problem of differential approximation [27].

Interim summary: So far several recurrent neural network architectures relevant to identification and control problems have been presented in a continuous-time state-space representation. The approximate realization of dynamical systems was discussed. Parallel and series-parallel identification models incorporating neural networks were introduced. The role of a tapped delay line in connection with static neural network architectures was investigated. It was proven that a static two-layer feedforward neural network with a tapped delay line can approximate a dynamical system to any degree of accuracy. Several further theorems concerning the approximation of dynamical system via recurrent neural networks were given. The results stated above are mainly existence results and do not provide constructional procedures for the network topology, its nonlinear quantities or even how to achieve the optimal parameter settings. The latter shall be further discussed in the next section.

NEURAL SYSTEM IDENTIFICATION

In order to understand the overall view of an identification problem, a general setup of the neural system identification procedure is introduced. The relevance and applicability of a neural network model in this setup will be discussed. The identifiability question is addressed for a fairly general class of recurrent neural networks. A major part of this manuscript consists of the presentation of model parameter estimation, also known as learning rules in the neural network context, based on results in optimization and Lyapunov stability theory. The improvement of stability and convergence results gained by the application of stability theory are pointed out.

The capability of a neural network architecture to model the behavior of a large class of dynamical systems within a certain accuracy is a basic requirement for the application of the neural network to the identification and control of nonlinear dynamical systems. The application of several recurrent neural models to a general dynamical system was established in the previous sections. In particular, only models that incorporate feedback connections will be valuable in identification of dynamical systems. In order to complete the identification procedure, now that several appropriate identification models are accessible, adaptation laws for the parameters of the model must be found; such learning laws must guarantee that the response of the model to an input signal indeed accurately approximates the output response of the real system. While the previous sections mainly presented existence results, the construction procedure of the neural models, concerning choice of values of model parameters shall now be discussed.

System identifiability: formal aspects [46]

The system identification procedure consists of first choosing either an appropriate identification model, or better, a model set. The parameters of this model are then adjusted according to an adaptive law such that the response of the model to an input signal approximates the output response of the real physical system to the same input. The latter is an estimation procedure that selects that member of the model set that appears to be most suitable for the purpose in question; one picks the "best" model in the given set. The selection is mostly performed in an iterative manner, guided by prior information and the outcomes of previous estimation attempts. Recall that, for example, the recurrent neural network model (20) is specified by a triplet of matrices (A,B,C) and the nonlinear function h. This will be denoted by

$$\Sigma = \Sigma_h(A,B,C) \qquad (56)$$

where A is a stable matrix, B is the weight parameter matrix and C is the output matrix. Assume that the structure of the neural model has been chosen according to some given identification, i.e. the nonlinear function h and the elements of the matrices A and C have been selected and the size of the matrix B is fixed. Notice that the weight parameters are still to be adjusted for a specific problem. Let θ be the parameter vector consisting of all adjustable parameters, d, in a general neural model. In the example, θ will contain all the elements of the matrix B. Now denote the general recurrent neural model by its dependence on the parameter vector

$$\Sigma = \Sigma(\theta) \qquad (57)$$

The following definitions were introduced for general dynamical systems in [46]. They are adapted here to the recurrent neural network context.

Definition (Neural Model Set): Let D_Σ be an open subset of $\Re^d$. A neural model set Σ^* is defined as

$$\Sigma^* = \left\{\Sigma(\theta) \middle| \theta \in D_\Sigma \subset \Re^d\right\}. \qquad (58)$$

The model set is clearly uncountable. Since a search for the "best" model has to be conducted over the entire model set, one can identify the parameter vector θ as a "smooth" index and perform the search over the parameter set (the index set). In fact, the presented neural models $\Sigma(\theta)$ are differentiable with respect to θ. Thus the following definition makes sense:

Definition (Neural Model Structure): A neural model structure Σ is a differentiable mapping from a connected, open subset D_Σ of $\Re^d$ to a neural model set Σ^*

$$\Sigma: \theta \in D_\Sigma \rightarrow \Sigma(\theta) \in \Sigma^*. \qquad (59)$$

Therefore with Σ, a certain model structure is selected which determines particular models $\Sigma(\theta)$ parametrized by the vector $\theta \in D_\Sigma \subset \Re^d$. This set of models is defined by EQ (58). Assume a set of data vectors from the real system is measured and given by:

$$z^N = \left[y(1), u(1), y(2), u(2), \ldots, y(N), u(N)\right], \quad (60)$$

where u and y denote system inputs and outputs respectively. Now, the problem is how to use the information contained in the data vector z^N to select a proper value θ^* of the parameter vector in D_Σ and hence a proper member $\Sigma(\theta^*)$ in the model set Σ^* most suitable to the purpose in question. This selection mechanism is defined as follows:

Definition (Parameter Estimation Method) : A mapping

$$z^N \rightarrow \theta \in D_\Sigma \quad (61)$$

is a parameter estimation method or learning algorithm.

In the neural network context a parameter estimation method is generally referred to as a learning algorithm.

Remark 10: Model structures are denoted by Σ above, whereas a particular model corresponding to the parameter vector θ is denoted by $\Sigma(\theta)$. Such a parametrization is instrumental in conducting a search for the "best" model. In general two different philosophies may guide the choice of parametrized model sets. First there are so-called *black-box* model structures. They represent flexible model sets that work with many different physical systems and in particular without detailed knowledge of the internal structure or physical descriptions of the modeled system. On the other hand, there are model structures with physical parameters. It is intended to incorporate physical insight into the model set. The chosen adjustable parameters will make sense physically and their number will be close to the actual, unknown number of parameters of the real system. The presented neural model sets clearly belong to the former black-box models.

The concept of identifiability addresses the problem of whether the identification procedure will yield a unique value of the parameter θ, whether different values of θ can give the same model and/or whether the resulting model is equal to the true system. Previous work in this area has been performed both for feedforward networks [58] as well as for recurrent networks [59] [60]; we shall review the implications of their work in the next subsection. In mathematical terms the former asks if the model structure Σ is one-to-one. In general there exist infinitely many parametric models $\Sigma(\theta)$ and some of these may not be identifiable. Define identifiability as follows [46]:

Definition (Identifiability): A model structure Σ is globally identifiable at θ^* if

$$\Sigma(\theta) = \Sigma(\theta^*), \ \theta \in D_\Sigma \Rightarrow \theta = \theta^*. \quad (62)$$

A model structure Σ is globally identifiable if it is globally identifiable at almost all $\theta^* \in D_\Sigma$.

If the neural model structure is one-to-one then it is identifiable. Identifiability also involves the question of whether the obtained data set is informative enough to distinguish between different models. The properties of the data used in parameter estimation are in fact crucial to the quality of the estimates. What properties must the input have to sufficiently excite all essential dynamics of the plant? In order to characterize process inputs the notion of persistent excitation has been introduced in the identification and adaptive control literature (such as in [46], [47] or [61]). In particular, conditions have been derived for the input signals of systems with a linear parametrization of the dynamics that guarantee the convergence of the model parameters to those of the true system.

It is important to point out that existing adaptive techniques for nonlinear systems generally require a linear parametrization of the plant dynamics, i.e. the parametric uncertainty is expressed linearly in terms of a set of unknown parameters. Then either through empirical studies or under this parameter-linearity assumption the stability and output error convergence are shown (e.g., see [22], [52], [62-63], [64], [65]). If the parameters are nonlinearly incorporated in the system dynamics then the question of persistent excitation is still an active current research problem. The reader is referred to the cited literature for a detailed discussion of the persistent excitation problem. At this point it shall be assumed that a persistently exciting input signal is available.

Remark 11: A significant ability of neural network models is generalization. A network generalizes if it is able to extrapolate information from a given set of training data to new inputs, from the same information domain, and responds with an output arbitrarily close to the corresponding output of the real system. In order to guarantee the generalization ability of the neural model, it has to be properly excited. Only careful choice of the set of training data can satisfy this requirement.

Input/Output Equivalence Approach to Identifiability

At this point it is clear that before actually designing an estimation algorithm it is necessary to verify whether there exists a unique set of values for the parameters of a recurrent neural model that realizes a given input/output mapping at all. Albertini and Sontag successfully addressed this problem in [59] for a class of recurrent neural models. They showed that there is only one way to build a network that achieves the design objective for a given input/output behavior. In particular, the neural "structure (weights) is uniquely determined by function (desired i/o behavior)" and therefore the weights of a continuous-time recurrent neural network are uniquely identifiable from input/output measurement. The following results are obtained from [8] and adapted to the above definitions. The approach is based on the definition of input/output (i/o) equivalence.

Definition (Ψ -I/O Equivalence): Let two general neural parametric models $\Sigma(\theta)$ and $\overline{\Sigma}(\overline{\theta})$ be given by the model structures Σ and $\overline{\Sigma}$ and be dependent on the parameter vectors θ and $\overline{\theta}$ respectively. Assume there exists a

diffeomorphism Φ between the state $\bar{x}$ of the model $\bar{\Sigma}(\bar{\theta})$ and the state x of the model $\Sigma(\theta)$ defined on a subset of $\Re^N$ such that $x = \Phi(\bar{x})$. Then $\Sigma(\theta)$ is Ψ-i/o equivalent to $\bar{\Sigma}(\bar{\theta})$ on $D_{\bar{\Sigma}}$ if and only if the input/output functions of $\Sigma(\theta)$ and $\bar{\Sigma}(\bar{\theta})$ are identical and there exists a diffeomorphism Ψ on $D_{\bar{\Sigma}}$, such that $\theta = \Psi(\bar{\theta})$.

Lemma 1 [59]: If $\Sigma(\theta)$ and $\bar{\Sigma}(\bar{\theta})$ are Ψ-i/o equivalent on $D_{\bar{\Sigma}}$ as the domain of Ψ then the model structure Σ is globally identifiable.

The proof of the lemma is straightforward and can be found in the reference [59]. Now consider two recurrent neural models $\Sigma(A,B,C)$ [††] and $\bar{\Sigma}(\bar{A},\bar{B},\bar{C})$, whose units have the same nonlinear activation function h, similar to EQ (11) of the form:

$$\begin{aligned} \dot{x} &= -x + \bar{y} + u, \quad x(0) = x_0, \\ \bar{y} &= H(Ax), \\ y &= Cx, \end{aligned} \tag{63}$$

where the nonlinear operator H is defined as above. The following assumptions are made:

Assumption 1: A is invertible; $\Sigma(A,B,C)$ is controllable and observable in the sense of linear systems, i.e. $[B,AB,\ldots,A^{n-1}B]$ and $[C^T,(CA)^T,\ldots,(CA^{n-1})^T]^T$ are nonsingular [66]; all elements of AB are non-zero and no two elements have the same absolute value;

Assumption 2: h is analytic; $h'(\lambda) \to 0$ as $\lambda \to \infty$; $h(0) = 0$, $h'(0) \neq 0$, $h''(0) = 0$ and $\exists\, k > 2$ such that $h^{(k)}(0) \neq 0$.

The following theorem due to Albertini and Sontag is stated here without proof.

Theorem 6 [59]: Let the two systems $\Sigma(\theta) = \Sigma(A,B,C)$ [‡‡] and $\bar{\Sigma}(\bar{\theta}) = \bar{\Sigma}(\bar{A},\bar{B},\bar{C})$ of type (63) be given which are of the same order N and satisfy the Assumptions 1-2. Then $\Sigma(\theta)$ is Ψ-i/o equivalent to $\bar{\Sigma}(\bar{\theta})$ iff there exists an invertible matrix T, such that $x = T\bar{x}$, and Ψ is defined by:

$$\begin{aligned} \bar{A} &= T^{-1}AT, \\ \bar{B} &= T^{-1}B, \\ \bar{C} &= CT, \end{aligned} \tag{64}$$

and

(i) if h is odd then $T=PD$, where P is a permutation matrix, i.e. its columns are permuted vectors of the canonical Euclidean basis, and

$D = diag\{\lambda_1,\ldots,\lambda_N\}, \lambda_i = \pm 1$ for all $i \in \{1,\ldots,N\}$,

(ii) if h is not odd then $T=P$.

By using Lemma 1, Theorem 6 proves global identifiability of the recurrent neural model (63). Thus the input/output behavior uniquely determines the weights, except for a reordering of the variables and, for odd activation functions, sign reversals of all incoming and outgoing weights at some units. In other words, if two recurrent networks have equal behaviors as black-box models then necessarily they must have the same number of units and, except at most for sign reversals at each node, the same weights.

The result for feedforward networks, due to [58], quoted from [65], is as follows: feedforward neural nets with a single hidden layer, a single output node and tanh(.) for for activation function, the neural network is uniquely determined by its I/O map, up to an obvious finite group of symmetrices (permutations of the hidden nodes and sign changes of all the weights associated with a particular hidden node), provided that the net is irreducible, i.e. there does not exist an inner node that makes a zero contribution to the output, and there is no pair of hidden nodes that could be collapsed to a single node without the I/O map.

The above results are closely related to the geometrric theory of nonlinear systems, and in particular to nonlinear realization theory [67].

Parameter estimation methods (learning algorithms) for neural networks

After the approximation properties of neural network architectures are known and the identifiability problem is clarified and established for a class of neural models, this subsection will present the estimation of the unknown network parameters. Parameter estimation methods are known as learning or training algorithms in the neural network community (learning refers to the biological origin of neural networks and their significant role in the field of artificial neural networks). On the other hand, from the systems theory viewpoint, a learning algorithm is just a parameter estimation method. In fact, many techniques employed in the field of neural networks have also been developed by the controls and optimization communities. In the sequel both notions will be used interchangeably.

A neural network model is determined by its network structure, its unit characteristics and the learning algorithm for its parameters. As presented in sections 2 through 4, neural models consist of many simple computational units which operate in parallel. The weight parameters define the strength of connection between the units. They are adapted during use in order to yield optimal performance. Given a set of inputs and desired outputs of a physical system, an appropriately chosen and trained neural model can emulate

[††] Notice that the dependence on the nonlinear function h is omitted.

[‡‡] Note that the vector θ contains all parameters of the system, i.e. the elements of the matrices A, B and C.

the mechanism which produced the data set. This subsection presents parameter estimation methods for several neural network models which are applicable in function approximation as well as system identification. The discussion covers the two branches of estimation methods which are based on optimization theory and Lyapunov stability theory. While the former shall only be briefly reviewed, the main emphasis is on the application of stability theory. The questions of stability and convergence will be important issues.

Estimation algorithms based on optimization theory

Parameter estimation methods for nonlinear models based on nonlinear programming techniques are well known. They can be found in many classical textbooks about optimization theory, statistics, identification, and adaptive control such as [46], [68], [47], and [69,70]. A very lucid review of available learning algorithms for feedforward neural networks and radial basis function networks can be found in [6].

A class of learning algorithms known as prediction error algorithms can be derived for feedforward neural networks by adopting ideas from nonlinear system identification [4]. The off-line version of the algorithm such as that presented in [5] uses a quadratic form of the prediction errors e_j of the available data (60) as an optimization criterion (i.e., the cost function) given by:

$$J(\theta) = \frac{1}{2N} \sum e_j^T(\theta)\, e_j(\theta) \qquad (65)$$

where θ is the parameter vector. The nonlinear programming problem is to find the optimum selection of parameters θ that minimize $J(\theta)$. Its minimization is usually achieved iteratively according to parameter adaptation rule:

$$\theta^k = \theta^{k-1} + \alpha \Xi(\theta^{k-1}) \quad (66)$$

k denotes the iteration index, α is a positive design constant (step size) and $\Xi(\theta)$ is the search direction. The search or adaptation direction $\Xi(\theta)$ of the parameters is based on information about the cost functional acquired at a previous iteration. The most widely used on-line method is based on the recursive implementation of the least-squares algorithm commonly using the negative gradient vector as the search direction. This algorithm is also known as the steepest descent learning rule. It guarantees the convergence to at least a local minimum of a convex cost functional. While the convergence rate is rather slow, the algorithm can be locally integrated into the parallel structure of feedforward neural networks. In order to improve the efficiency of minimization, the negative gradient direction has been modified in other approaches. The Gauss-Newton algorithm, also known as the full prediction error algorithm, utilizes the Hessian matrix of the cost functional in the adaptation law and results in a very efficient convergence. Unfortunately, it requires great computational power and has a centralized structure when applied to neural networks. Moreover, the size of the Hessian leads to an enormous memory demand for large networks. However, these limitations can be overcome by the parallel prediction error algorithm described in [5]. Less frequently the search direction is computed via conjugate-gradient, Newton, quasi-Newton or other variations of first and second order methods. The step size α must be appropriately chosen to guarantee the convergence of the iterative procedure; line minimization, Armijo and other inexact methods have been utilized in the past. The off-line prediction error algorithms typically utilize gradient-descent optimization techniques and are guaranteed to converge to a local minimum that contains the initial parameter vector in its basin of attraction. The corresponding on-line parameter estimation methods include: recursive steepest descent or smoothed stochastic gradient algorithm, full recursive prediction and parallel recursive prediction error algorithm. It can be shown that the recursive algorithms have the same convergence properties as their batch counterparts.

The famous back propagation learning algorithm for multilayer feedforward neural networks first introduced by [71] and made popular later by [9] is a recursive approximation of the steepest-descent algorithm. As mentioned, the main strength of the back propagation algorithm lies in its computational simplicity and parallel structure in which learning is distributed throughout all units. On the other hand, slow convergence to local minima are its disadvantages. Nevertheless, the backpropagation learning algorithm has proven very successful in empirical studies (for example [12], [13]).

Several training methods have been proposed for recurrent neural networks and most of them again rely on gradient methods. They are more or less extensions of the back propagation algorithm for feedforward neural networks. [72]] introduced recurrent back propagation as a fixed-point learning algorithm, i.e. the recurrent neural model learns equilibria or steady states in a sense related to a content addressable memory (CAM) for which Hopfield used the Hebbian learning rule ([38], [39] and [40]). Werbos designed an off-line adaptation mechanism, back propagation though time, in [73] that can learn time trajectories. [20] developed a real-time recurrent learning algorithm which can solve the trajectory learning problem on-line. The dynamical back propagation algorithm introduced in [23] relies on sensitivity models. All these algorithms have been successfully applied in the identification and control of highly uncertain, nonlinear and complex systems[§§]. In spite of the successful studies there are some major drawbacks of the gradient-based approaches, including the great computational effort, the need of global information because the learning is not localized in single units, and most importantly the inability to theoretically derive satisfactory stability and convergence results when the adaptation is carried out on-line. In particular, on-line adaptation is a major concern in neural identification and control and will be further addressed below.

[§§] For collections of papers on neurocontrol see for example [12-18].

In order to demonstrate the gradient descent technique, an on-line back propagation learning algorithm for feedforward distributed neural networks (introduced in subsection 4.5) was derived [34,36-37]. The network contained neural with internal feedback inside the units; these networks are placed among the recurrent neural models. This approach avoids the need for a tapped delay line and therefore is considerably faster than time-delay versions of static networks. We utilized such networks in the approximation of sinusoids, in computer simulations involving approximation and control of a Boeing-727 aircraft passing through windshear, and in control of an S-10 Blazer with wind and road disturbances [28-37]. In using that identification approach, one needs to make only minute assumptions as to the structure of the mathematical model. In general a detailed description of the plant is assumed to be unknown. The linear models replace real physical systems and generate the necessary input/output data for the training. The plant models are more or less arbitrary and highly nonlinear. Of course, the stable oscillating input/output behavior of the given models is very convenient in order to demonstrate the improved identification of the system dynamics.

Estimation based on stability theory

Gradient learning techniques have been efficiently used as parameter adaptation mechanisms in practice, but problems have often been encountered in proving their stability in closed-loop identification and control setups. This problem is even more severe if the adaptation is carried out on-line. The convergence of the adaptation mechanism to a global minimum can only be assured in the case of a convex error surface; this is a very limiting assumption. In practice, problems are characterized by very "lumpy" error surfaces; the convergence to a global minimum and thus the convergence of the output error to zero can hardly be achieved.

In order to overcome the stability and convergence problems a group of researchers have recently avoided iterative training procedures in favor of provable stable adaptation techniques. The approach was previously discovered in robust adaptive control and is now utilized for tuning of the neural network model parameters: the Lyapunov synthesis approach (see for example [74] or [47]). These developments have enhanced the understanding of neural on-line parameter estimation in the context of closed loop dynamical systems by providing a link to adaptive control theory. With a Lyapunov-like synthesis approach, a dynamical equation is first obtained in terms of the error signal which includes both the estimation and parameter error. A certain Lyapunov-like function V is then considered whose time derivative $\dot{V}$ along the trajectories of the dynamical system equation is made non-positive by properly designing the adaptive law for the adjustable parameters. The properties of V and $\dot{V}$ are then used to establish the stability properties of the on-line estimation scheme.

The discussion shall be continued with the derivation of a neural learning algorithm for higher-order recurrent neural networks presented in subsections 4.3 and 4.4. These are based on Lyapunov's direct method (see for example [57]). It demonstrates the procedure for developing the dynamical error equation and choosing an appropriate Lyapunov-like function V. The results were published in [45] and successfully applied in the identification of a nonlinear robotic manipulator.

Learning algorithm for RHONN:
Filtered Error Identification Model

The modeling error plays a crucial role in the design of the learning algorithm for recurrent neural network as identification models. The modeling error is the mismatch between the real system and the neural network with "optimally" chosen weight parameters. In general there will always arise modeling inaccuracies that are mainly due to an insufficient number of adjustable weights. Although not very realistic, the first approach presented will assume a zero modeling. In order to avoid a parameter drift caused by modeling errors, the algorithm will be revised such that the learning is robust. Given the general nonlinear system:

$$\dot{x} = f(x,u), \quad x(0) = x_0 \qquad (67)$$

the existence of a higher order recurrent neural network that approximates (67) is guaranteed by Theorem 5. Recall the recurrent neural model (27):

$$\begin{aligned} \dot{x} &= Ax + Bg(x,u), \quad x(0) = x_0, \\ y &= H(x^L), \end{aligned} \qquad (68)$$

where B is the adjustable weight parameter matrix. If one assumes that there is no modeling error, then by Theorem 5 there exists an optimal weight matrix B^* such that the unknown dynamical system (67) is exactly rendered by the dynamical state equation:

$$\dot{x} = Ax + B^* g(x,u), \quad x(0) = x_0, \qquad (69)$$

Note that if by assumption the input *u(t)* and the state *x(t)* are bounded for all $t \geq 0$ then $g(x,u)$ is also bounded.

Remark 12: Due to the identifiability discussion earlier, if the optimal weights are not unique then B^* denotes an arbitrary but fixed element of the set of optimal weight parameter matrices.

Now, assume that the state x is available for measurement and choose the neural identifier as:

$$\dot{\hat{x}} = A\hat{x} + Bg(x,u), \quad \hat{x}(0) = \hat{x}_0, \qquad (70)$$

where B is the estimate of the unknown weight parameter matrix. Introducing the weight error matrix $\tilde{B} = B - B^*$ and the dynamics of the state error $\tilde{x} = \hat{x} - x$, the system equations are described by:

$$\dot{\tilde{x}} = A\tilde{x} + \tilde{B}g(x,u), \quad \tilde{x}(0) = \hat{x}_0 - x_0. \qquad (71)$$

In order to derive a stable adaptation law for the weight parameters consider the Lyapunov function candidate:

$$V(\tilde{x},\tilde{B}) = \frac{1}{2}\left(\tilde{x}^T\tilde{x} + \frac{1}{\gamma}\,trace\{\tilde{B}\tilde{B}^T\}\right), \quad (72)$$

where $\gamma \geq 0$ is a design constant. Recalling EQ (19) (for the weight matrix, with $N=n$, the derivative of V w.r.t. t is:

$$\begin{aligned}\dot{V}(t) &= \tilde{x}^T\dot{\tilde{x}} + \frac{1}{\gamma}\sum_{i=1}^{n} b_i^T\dot{b}_i \\ &= \tilde{x}^T A\tilde{x} + \tilde{x}^T B g(x,u) + \frac{1}{\gamma}\sum_{i=1}^{n} b_i^T\dot{b}_i \\ &= \tilde{x}^T A\tilde{x} + \sum_{i=1}^{n}\left[b_i^T g(x,u)\tilde{x}_i + \frac{1}{\gamma}b_i^T\dot{b}_i\right].\end{aligned} \quad (73)$$

Notice in the first step above that B^* in $\tilde{B}$ is fixed and thus $\dot{\tilde{B}} = \dot{B}$. Now it is clear that if choosing

$$\dot{b}_i = -\gamma\, g(x,u)\,\tilde{x}^T, \qquad i = 1,\ldots,n, \quad (74)$$

or in matrix from:

$$\dot{B}^T = -\gamma\, g(x,u)\,\tilde{x}^T, \quad (75)$$

and since A is a Hurwitz matrix as defined in EQ (18), then

$$\dot{V} = \tilde{x}^T A\tilde{x} \leq 0. \quad (76)$$

Regarding EQ (72) this implies that $\tilde{x}$, $\tilde{b}_i = b_i - b_i^* \in L_\infty$, i.e. $\tilde{x}$ and $\tilde{b}_i$ $(i = 1,\ldots,n)$ are uniformly bounded. Furthermore, from (71) with the remark above that $g(x,u)$ is bounded it also follows that $\dot{\tilde{x}} \in L_\infty$. Moreover, since V is a nonincreasing function of time and bounded from below, there exists a finite limit V_∞ such that

$$0 \leq V_\infty = \lim_{t\to\infty} V(t) \leq V(0). \quad (77)$$

Since A is symmetric, the derivative of $\dot{V}$ is:

$$\ddot{V} = 2\tilde{x}^T A\dot{\tilde{x}}, \quad (78)$$

with $\tilde{x}$ and $\dot{\tilde{x}} \in L_\infty$ then $\ddot{V} \in L_\infty$, i.e. $\ddot{V}$ is uniformly bounded. The boundedness of $\ddot{V}$ implies that $\dot{V}$ is uniformly continuous in time. The application of Barbalat's Lemma then indicates that $\dot{V}(t) \to 0$. Hence by (76) $\tilde{x}(t) \to 0$. Therefore the adaptation law (75) guarantees that the identification error converges to zero. This result shall be summarized in the following theorem:

Theorem 7 (Filtered Error Model Learning): Consider the neural identifier

$$\dot{\hat{x}} = A\hat{x} + Bg(x,u), \quad \hat{x}(0) = \hat{x}_0, \quad (79)$$

whose weights are adjusted by the adaptation law

$$\dot{B}^T = -\gamma\, g(x,u)\,\tilde{x}^T, \quad (80)$$

where γ is the adaptation rate. Then, under the assumption of no modeling error, the neural identification scheme guarantees the following properties:

(i) $\tilde{x}$ and $b_i \in L_\infty$,

(ii) $\lim_{t\to\infty} \tilde{x}(t) \to 0$.

Proof of Theorem 7 follows directly from the previous discussion and derivations.

Remark 13: Although the error $\tilde{x}$ converges to zero as the time t goes to infinity, the result does not imply that the error dynamics are not asymptotically stable.

Remark 14: The above adaptation law does imply that the weight estimation error $\tilde{B}$ is bounded but does not indicate by any means that this error converges to zero. As mentioned earlier, in order to achieve convergence of the parameters to their correct value the additional condition of persistent excitation must be satisfied by the nonlinear regressor function $g(x,u)$. The signal $g(t) = g(x(t),u(t))$ will be persistently exciting if there exist positive constants α_1, α_2 and T such that for all $t \geq 0$

$$\alpha_1 I_N \leq \int_t^{t+T} g(\tau)\, g^T(\tau)\, d\tau \leq \alpha_2 I_N. \quad (81)$$

Then in the absence of modeling errors the adaptive control law without modification (see below) can also guarantee parameter convergence, i.e.

$$B(t) \to B^* \quad as \quad t \to \infty. \quad (82)$$

Obviously for nonlinear systems this condition cannot be verified *a priori*.

Remark 15: Most of the existing adaptive techniques for nonlinear systems require a linear parametrization of the plant dynamics, i.e. the parameters linearly enter the nonlinear dynamical equation (see also [75]). In fact, the higher-order recurrent network model (70) is linearly parameterized in the adjustable weight matrix.

Remark 16: An adaptive neural identifier of the form (70) is known as filtered error identifier. This can be seen from the first-order stable filter depicted in Figure 4 with $A = -diag\{a,\ldots,a\}$. The error filtering design is discussed in detail in [75]. Once the model parameters are successfully trained, the model input x can be replaced by its estimate $\hat{x}$ and the training can be stopped. The criteria for a successful training are critical, in particular if the convergence of the parameters to their optimal values cannot be guaranteed.

Robust Learning Algorithm For Recurrent Higher-Order Neural Networks (RHONN)

The assumption of no modeling error between the physical plant and the neural network model in the filtered error recurrent higher order neural network training algorithm is crucial. The modeling error is mainly caused by an insufficient number of higher-order terms in the neural model. In order to accommodate for modeling inaccuracies which can result in a parameter drift*** i.e. the weight parameters drift to infinity and the estimation error diverges, the presence of modeling errors will be allowed. The modeling error will appear as an additive disturbance in the

*** Parameter drift is an instability problem that occurs in on-line adaptation and is very well known in adaptive control theory.

differential equations representing the system. A robust learning algorithm is presented next.

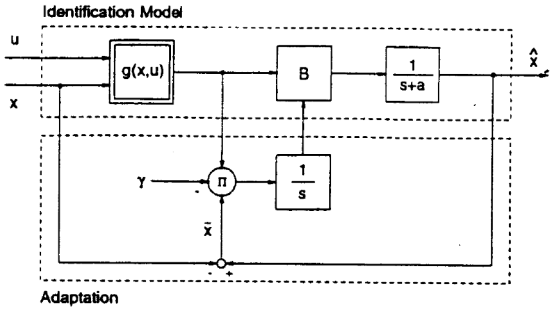

Figure 4: Filtered error identification model

Given the nonlinear system (67). By adding and subtracting the term $Ax + B^* g(x,u)$ the system can be restated by:

$$\dot{x} = Ax + B^* g(x,u) + v \quad (83)$$

where the modeling error v is given by

$$v = f(x,u) - Ax - B^* g(x,u). \quad (84)$$

Note that $v \in L_\infty$ because x and u are assumed to be bounded. Let K be a compact subset of $\Re^{n+m}$. Since by assumption the system to be identified is bounded-input bounded-output (BIBO) stable and u is uniformly bounded in time, assume that K is chosen such that $(x,u) \in K$ for $t \geq 0$. The optimal weight parameter matrix B^* minimizes the modeling error v on K in the sense of the L_∞-norm, subject to the constraint $\|b_i\| \leq M_i$, where M_i is a large design constant, and the optimal weights for the RHONN are:

$$b_i^* = \arg \min_{\|b_i\| \leq M_i} \sup_{(x,u) \in K} |v_i|. \quad (85)$$

Thus a bound on the network weights is introduced which avoids numerical problems due to very large weight values and allows the adaptation law to be modified by adding a leakage term known as σ-modification from adaptive control (see [74]) that prevents the weight values from drifting to infinity. Choosing the neural identifier as (70), the identification error dynamics are given by

$$\dot{\tilde{x}} = A\tilde{x} + \tilde{B}g(x,u) - v. \quad (86)$$

The next theorem is stated from [45].

Theorem 8 (Robust Filtered Error Model Learning, [45]): Consider the neural identifier

$$\dot{\hat{x}} = A\hat{x} + Bg(x,u), \quad (87)$$

whose weights are adjusted for $i = 1,\ldots,n$ as follows

$$\dot{b}_i = \begin{cases} -\gamma\, g(x,u)\, \tilde{x}_i, & \|b_i\| \leq M_i \\ -\gamma\, g(x,u)\, \tilde{x}_i - \sigma\gamma b_i, & \|b_i\| > M_i \end{cases} \quad (88)$$

where γ is the adaptation rate and $\sigma > 0$ is a design constant for the leakage term $-\sigma\gamma b_i$. Then the robust neural identification scheme guarantees the following properties:

(i) $\tilde{x}, b_i \in L_\infty, i = 1,\ldots,n$,

(ii)] there exist constants λ, μ such that

$$\int_0^t \|\tilde{x}(\tau)\|^2 \, d\tau \leq \lambda + \mu \int_0^t \|v(\tau)\|^2 \, d\tau \quad (89)$$

The proof can be found in [45]. The above robust adaptation rule guarantees that the output error of each dynamical unit is bounded; and the state error is proportional to the energy of the modeling error. If the modeling error is square integrable then the state error converges to zero asymptotically. An immediate corollary of Theorem 8 is:

Corollary 1: If the modeling error in the neural identification scheme is zero, i.e. $v = 0$, then $\lim_{t \to \infty} \tilde{x}(t) = 0$.

Remark 17: These results require exact knowledge of neither the compact set K nor the upper bound of the modeling error v.

Remark 18: Since the σ-modification causes the adaptive law to be discontinuous, the existence and uniqueness results of solutions to the resulting differential equation are, in general, not applicable. Therefore [45] suggests a further modification that results in smooth weight parameter trajectories.

Remark 19: Notice that the modeling error (84) can also include a bounded disturbance signal. Assume that equation for the dynamical system is given by:

$$\dot{x} = f(x,u) + d(t), \quad x(0) = x_0, \qquad (91)$$

where d is a time-varying bounded disturbance signal, i.e. there exists a finite $B_d > 0$ such that for all t

$$\|d(t)\| \le B_d. \qquad (92)$$

Then the new modeling error $\bar{v}$ is given

$$\bar{v} = v + d. \qquad (93)$$

INTELLIGENT FLIGHT CONTROL

The intelligent flight control concept program [31] aims to develop and flight demonstrate a flight control system that can efficiently identify aircraft stability and control characteristics of the aircraft. Then utilize this information to optimize the performance of the aircraft. During the months of April - December 1995 we were involved with and worked on the joint NASA-Ames, MDA-St. Louis, TSU, and WU's Advanced Concept Program for Intelligent Flight Control. Our group at Washington University participated in Phase II, the online learning, with neural networks that learn new information during flight, utilizing recurrent higher order neural networks which were presented in earlier subsections. More specifically, a recurrent second-order neural network architecture with a robust filtered error learning algorithm was utilized to identify the dynamics of an F-15 aircraft.

The estimated flight coefficients and derivatives are fed into an optimal controller that maximizes the system performance. The new control scheme uses neural networks to monitor the operating characteristics of an aircraft on a real-time basis and can modify the flight control to accommodate various types of failures or damage. The flight sensor measurements and estimates of flight coefficients and derivatives derived from the motion equations are available for the training of the neural network. A first approach is the application of static pre-trained baseline neural networks that measure aircraft performance factors which can be used to provide optimal control. In particular, the baseline networks work as memory elements or look-up tables for flight coefficients and derivatives. Modeling the aircraft with other analytical methods would result in computer code that is overly large to efficiently function in real-time computer control of the aircraft. A second approach is the development of online neural networks that can learn new information during flight in order to estimate off-nominal values or provide model updates during failure and damage conditions.

While the emphasis of this work has been the development and implementation of online neural network estimators, results with and without the baseline network have been discussed. The system architecture which incorporates a baseline network is depicted in Figure 5.

Our task was to develop a candidate online neural network algorithm which processed aircraft sensor data in real-time and updated the aerodynamic coefficients for stability and control derivatives. The modeling problems to be solved by the neural network consist of first, to determine a correct aircraft model when an unpredicted event occurs which changes the critical stability and control properties of the aircraft, and second, to be able to update the existing neural-augmented aircraft model by using available sensor information to gain increased performance advantages for flight envelope expansion. In other words, the candidate neural network structure with appropriate learning algorithm will identify the time-varying matrices A and B of the linearized aircraft model $\dot{x}(t) = A(t)x(t) + B(t)u(t)$.

The estimated matrices A and B which contain flight coefficients and derivatives will be utilized in the computation of the controller gain of an optimal controller by a real-time Riccati-solver[†††] [32-33]. The candidate neural system has to be capable of recognizing temporal patterns or, more desirably, be able to fully identify the time-varying mapping between sensor values and matrices A and B. Since any time-varying mapping may be described by a dynamical system, the latter is in particular the identification of the dynamical system whose states are the flight coefficients and derivatives driven by the given sensor measurements. The adaptation of the network parameters will be performed online in order to allow the instantaneous adaptation to flight situations. Therefore the derivation of an efficient parameter estimation algorithm is necessary that guarantees stability and convergence of the overall system.

The earlier discussions justify the application of recurrent higher-order neural networks to the above system identification problem. In particular, a second-order recurrent neural network architecture was implemented in software in order to identify the critical flight coefficients and derivative values using the real-time robust filtered error learning algorithm presented in subsection 5.3.4. Identification results of a flight coefficient and its derivative values are depicted in Figures 6-9. The inputs to the networks are several measured sensor values of the aircraft. The output trajectories of the dynamical neural network are depicted as solid lines and the desired outputs are shown in dashed lines. In Figure 6 the dynamical neural network learns the deviations of the static baseline network outputs from the desired values which are provided by the motion equations as shown in Figure 5. The outputs of the baseline network are plotted with dotted lines. The dynamical network is continuously trained while the static baseline network functions in the sense of a look-up table. The same simulations without the use of the baseline network are depicted in Figure 7 (i.e, no baseline network is included and the initial learning begins with no *a priori* training). Figures 8-9 depict the on-line learning of an aircraft parameter with and without a baseline network; the graphs correspond to actual flight data and the corresponding relative error respectively. In these results we estimated 21 aircraft parameters; here we shall show the results for four such parameters, these are: *Cm* (pitching moment), *Cmdh* (partial derivative of the pitching moment with respect to

[†††] We implemented a real-time (< 20 msec.) Riccati solver for the optimal control of a damaged aircraft during Sept. 1994- August 1995 for McDonnell Douglas for the Intelligent Flight Control for the Fly-By-Light Advanced Hardware System [32-33].

stabilator), *Cmq* (partial derivative of the pitching moment w.r.t. pitch rate), and *Cma* (partial derivative of the pitching moment w.r.t. angle of attack, including Canard). The graphs provided here, correspond to an altitude of 30,000 ft. , a speed of Mach 1.2, and a response to a lateral stick input. The utilization of a baseline network reduces the initial estimation error of the online dynamical neural network and therefore overcomes the necessity to pre-train it with aircraft take-off data. For further information refer to the project report [29,31].

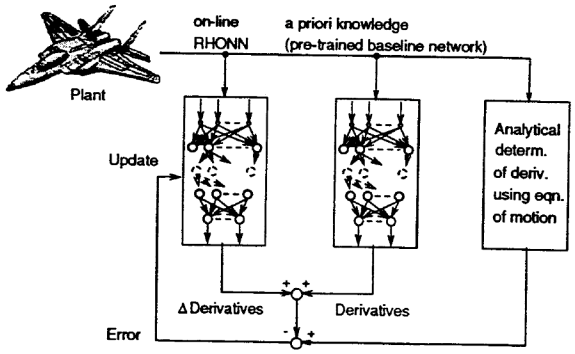

Figure 5: System architecture with incorporated baseline network

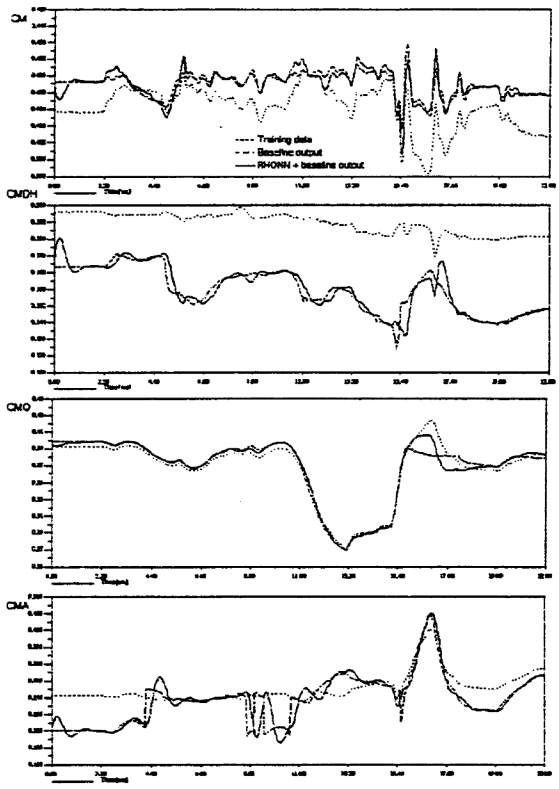

Figure 6: On-line recurrent second-order neural network identification of aircraft parameters with incorporated baseline network.

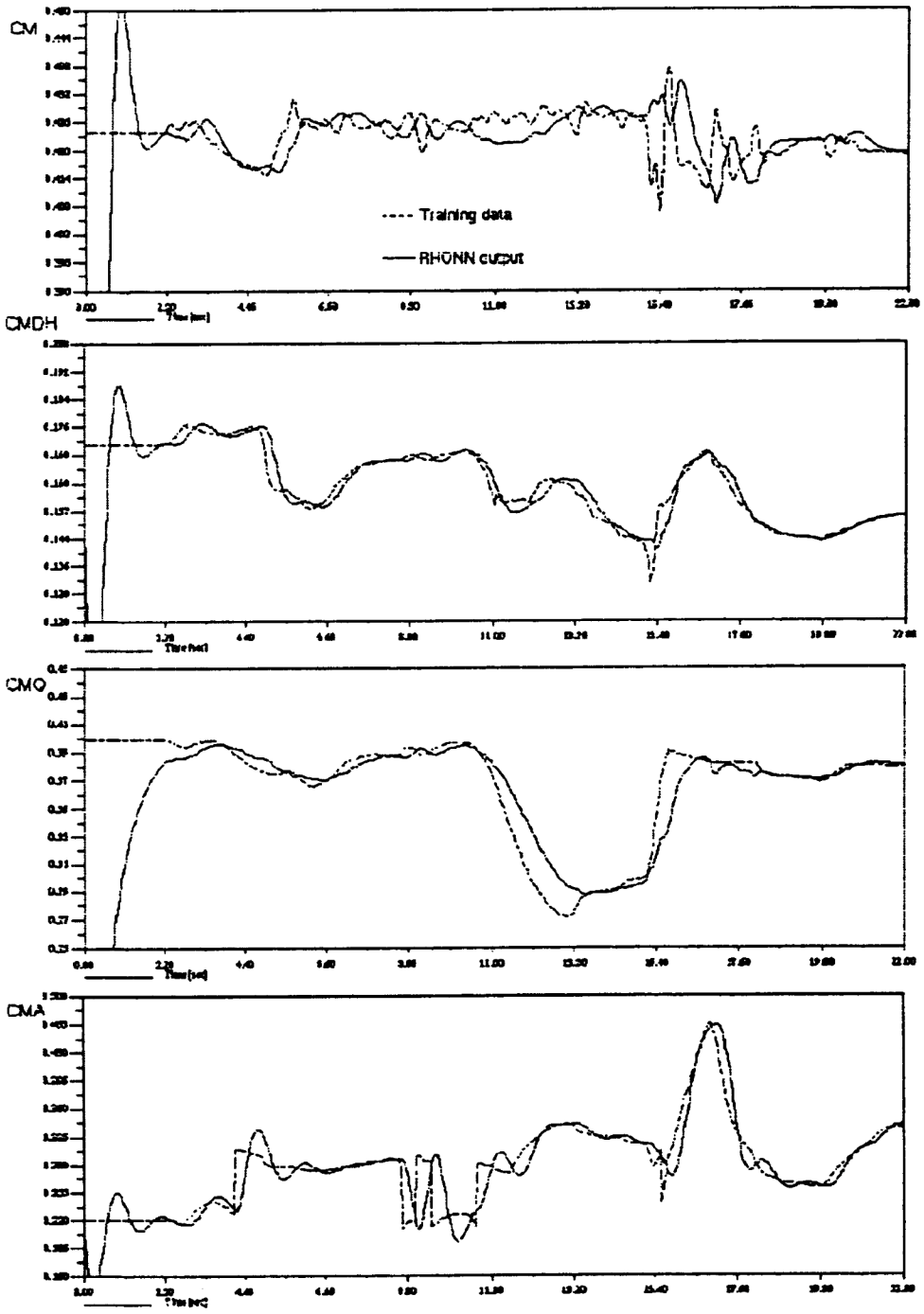

Figure 7: On-line recurrent second-order neural network identification of aircraft parameters without a baseline network

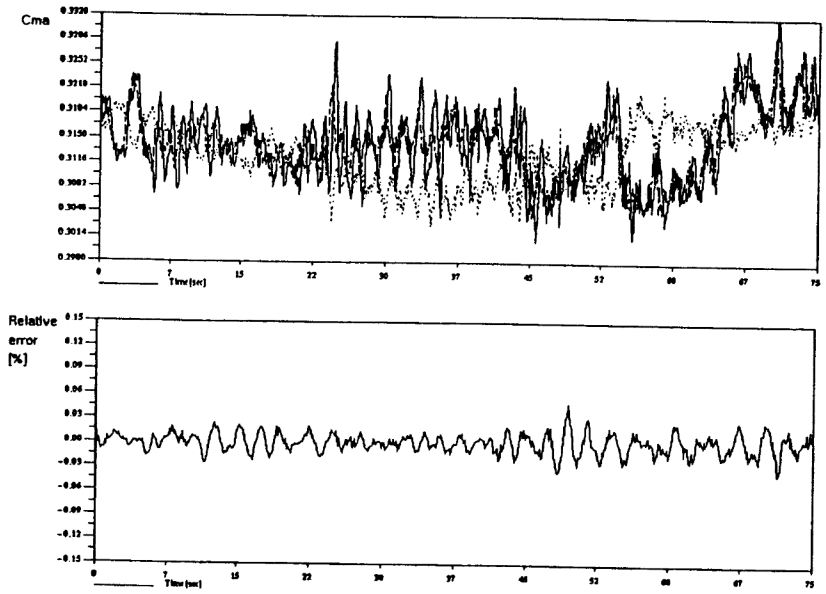

Figure 8: On-line learning of an aircraft parameter (Cma) with a baseline network

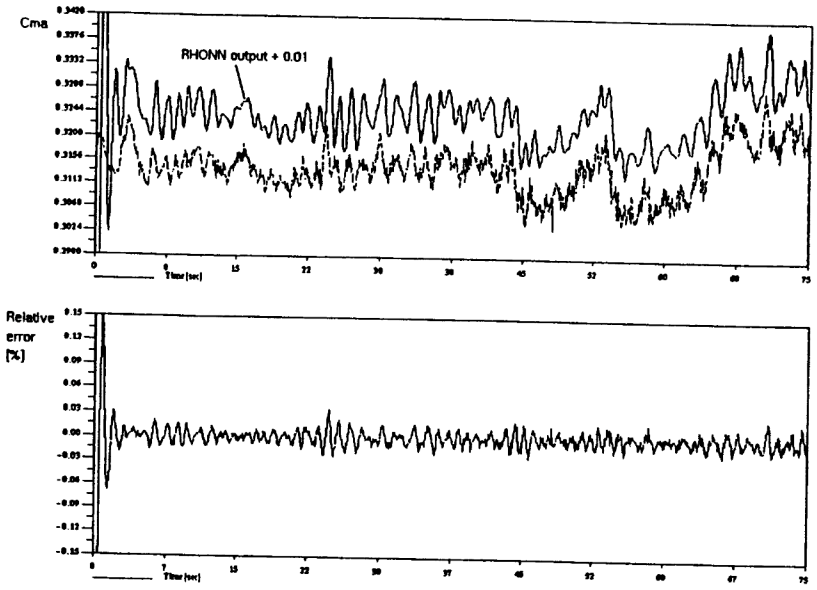

Figure 9: On-line learning of an aircraft parameter (Cma) without a baseline network

7.0 CONCLUSIONS

The neural identification problem was outlined in a general form applying several results from classical system identification to the neural network context. The identifiability of a general class of recurrent neural networks was established via the input/output equivalence approach by Albertini and Sontag. Several parameter estimation algorithms based on optimization and Lyapunov theory were presented. An on-line learning algorithm based on the classical backpropagation approach which had been derived for feedforward distribute dynamical neural networks was discussed. The algorithm allows the adjustment of the dynamical parameters and therefore significantly improves the behavior of the identification results for dynamical systems. The drawbacks concerning convergence and stability properties of gradient descent algorithms in on-line closed-loop systems were pointed out. An improved on-line parameter estimation algorithm based on a Lyapunov-like approach was presented that guarantees stability and convergence of the error to zero in the case of vanishing modeling. The influence of the neural modeling error on the performance of the estimation led to a modified algorithm which guarantees robustness. All algorithms were simulated; in particular, a recurrent second-order neural network architecture with a robust filtered error learning algorithm was utilized to identify the dynamics of an aircraft.

REFERENCES

[1] Leontaritis, I. and Billings, S. Input-output parametric models for non-linear systems. *International Journal on Control*, 41 (2), 303-328, 1985.

[2] Chen, S. and Billings, S. Representation of nonlinear systems: the narmax model. *International Journal on Control*, pp. 1013-1032, 1989.

[3] Billings, S. Identification of nonlinear systems- a survey. *IEE Proceedings*, Part D, 1980.

[4] Chen, S. and Billings, S. Recursive prediction error parameter estimator for nonlinear models. *International Journal on Control*, 49 (2), pp. 569-594, 1989.

[5] Chen, S., Cowan, C., Billings, S., and Grant P. Parallel recursive prediction error algorithm for training layered neural networks. *International Journal on Control*, 51 (6), pp. 1215-1228, 1990.

[6] Chen, S. and Billings, S. Neural networks for nonlinear dynamic system modelling and identification. *International Journal on Control*, 56 (2), pp. 319-346, 1992.

[7] Chen, T. and Chen., H. Approximation of continuous functionals by neural networks with application to dynamic systems. *IEEE Trans. on NNs*, 4 (6), pp. 910-918, 1993.

[8] Zbikowski, R., Hunt, K.J., Dzielinski, A., Murray-Smith, R., and Gawthrop P.J. A review of advances in neural adaptive control systems. Technical Report of the ESPRIT NACT Project TP-1, Glasgow University and Daimler-Benz Research.

[9] Rumelhart, D. and McClelland, J. *Parallel Distributed Processing,* Volume 1: Foundations. MIT Press, Cambridge, Massachusetts, 1986.

[10] Hecht-Nielsen, R. *Neurocomputing*. Addison-Wesley, 1990.

[11] Hertz, J., Krogh, A., and Palmer, R. *Introduction to the Theory of Neural Computation*. Addison-Wesley, 1991.

[12] Miller, W., Sutton, R., and Werbos, P., editors. *Neural Networks for Control*. The MIT Press, Cambridge, Massachusetts, 1990.

[13] White, D. and Sofge, D., editors. *Handbook of Intelligent Control: Neural, Fuzzy and Adaptive Approaches*. Van Nostrand Reinhold, New York, 1992.

[14] Gupta, M.M. and Rao, D.H., editors, *Neuro-Control Systems: Theory and Applications*, IEEE Press, 1994.

[15] Gupta, M.M. and Sinha, N.K., editors, *Intelligent Control Systems*, IEEE Press, 1996.

[16] Chen, C.H., editor, *Fuzzy Logic and Neural Network Handbook*. McGraw-Hill, New York, 1996.

[17] Anderson, J.A. and Rosenfeld, E., editors, *Neurocomputing: Foundations of Research*, MIT Press, 1989

[18] Anderson, J.A., Pellionisz, A., Rosenfeld, E. (Editors), *Neurocomputing: Directions forResearch*, MIT Press, 1990.

[19] Hunt, K., Sbaraboro, D., Zbikowski, R., and Gawthrop, P. Neural networks for control systems - a survey. *Automatica*, 28 (6), 1083-1112, 1992.

[20] Williams, R. and Zipser, D. A learning algorithm for continually running fully recurrent networks. *Neural Computation*, 1, 270-280, 1989.

[21] Narendra, K. and Parthasarathy, K. Backprogagation in dynamical systems containing neural networks. Technical Report 8905, Yale University, 1989.

[22] Narendra, K. and Parthasarathy, K. Identification and control of dynamic systems using neural networks. *IEEE Transactions on Neural Networks*, 1 (1), 4-27, 1990.

[23] Narendra, K. and Parthasarathy, K. Gradient methods for the optimization of dynamical systems containing neural networks. *IEEE Transactions on Neural Networks* , 2 (2), 252-262, 1991.

[24] Levin, A.U., and Narendra, K.S. Identification of nonlinear dynamical systems using neural networks. in *Neural Systems for Control*, O. Omidvar and D.L. Elliot (Efs.), pp. 129-160, Academic Press, 1997.

[25] Levin, A.U., and Narendra, K.S. Control of nonlinear dynamical systems using neural networks, part II. *IEEE Trans. on Neural Networks*, Vol. 7, pp. 30-42, January 1996.

[26] Levin, A.U., and Narendra, K.S. Control of nonlinear dynamical systems using neural networks. *IEEE Trans. on Neural Networks*, Vol. 4, pp. 192-206, March 1993.

[27] Zbikowski, R. *Recurrent Neural Networks: Some Control Aspects*. Ph.D. thesis, Glasgow University, 1994.

[28] Amin, S.M. and Gerhart, V., System Identification via Artificial Neural Networks. *IEEE Trans. on Automatic Control*, in review, July 1997.

[29] Amin, S.M., Rodin, E.Y. and Wu, A.Y. Neurocontrol of Nonlinear Systems via Local Memory Neurons. accepted for publication, Journal of *Mathl. Comput. Modelling*, Elsevier Sci. Ltd, MCM 2557, 26 pp., Fall 1997.

[30] Amin, S.M, Rodin, E.Y., and Wu, A.Y. Application of dynamic neural networks to approximation and control of nonlinear systems. *Proc. of 1997 American Control Conference*, pp. 222-226, Albuquerque, June 4-6, 1997.

[31] Amin, S.M, Rodin, E.Y., and Gerhart, V. Intelligent flight control system: system identification via dynamical neural networks. Final Report and the User's Manual for the RHONNI Simulator, submitted to McDonnell Douglas and NASA-Ames, Jan. 1996.

[32] Amin, S.M, Rodin, E.Y., and Chen, Y-M. Real-Time Riccati Solver; Vol. II: Implementation and Results. Final Report submitted to McDonnell Douglas Corp., Sept. 1995.

[33] Amin, S.M, Rodin, E.Y., and Chen, Y-M. Real-Time Riccati Solver; Vol. I: Theory and Algorithms. Final Report submitted to McDonnell Douglas Corp., Sept. 1995.

[34] Amin, S.M, Rodin, E.Y., and Wu, A.Y. Neurocontrol of an aircraft: application to windshear. *Mathematical and Computer Modelling* , Elsevier Sci. Ltd, Vol. 22, No. 1, pp. 63-78, May 1995.

[35] Amin, S.M, Rodin, E.Y., and Wu, A.Y. System identification and disturbance attenuation via dynamic neural nets. *Proc. of Neural Nets for Aero Control Symp.*, NASA Ames Research Center, CA, 16 pp., Aug. 1994.

[36] Wu, A.Y., Amin, S.M, and Rodin, E.Y. Control and disturbance rejection with a dynamic neurocontroller. *Intelligent Engineering Systems Through Artificial Neural Networks*, ASME Press, pp. 643-649, 1992

[37] De, P., Amin, S.M, and Rodin, E.Y. System identification with dynamic neural networks. *Intelligent Engineering Systems Through Artificial Neural Networks*, ASME Press, pp. 97-103, 1992..

[38] Hopfield, J. Neural networks and physical systems with emergent collective computational abilities. In *Proceedings of the National Academy of Science*, vol. 79, pp. 2554-2558, 1982.

[39] Hopfield, J. Neurons with graded response have collective computational properties like those of two-state neurons. In *Proceedings of the National Academy of Science*, volume 81, pp. 3088-3092, 1984.

[40] Hopfield, J. and Tank, D. Neural computation of decisions in optimization problems. *Biological Cybernetics*, 52, 141-152, 1985.

[41] Cohen, M. and Grossberg, S. Absolute stability of global pattern formation and parallel memory storage by competitive neural networks. *IEEE Trans. on SMC*, 13 (5), pp. 815-826, 1983.

[42] Peretto, P. and Niez, J. Long term memory storage capacity of multiconnected neural networks. *Biological Cybernetics*, 54, 53-63, 1986.

[43] Baldi, P. Neural networks orientations of hypercube and algebraic threshold functions. *IEEE Trans. on Information Theory*, 34 (3), 1988.

[44] Kamp, Y. and Hasler, M. *Recursive Neural Networks for Associative Memory*. John Wiley & Sons, 1990.

[45] Kosmatopoulos, E. Polycarpou, M., Christodoulou, M., and Ioannou, P. High-order neural network structures for identification of dynamical systems. *IEEE Transactions on Neural Networks*, 6 (2), 422-431, 1995.

[46] Ljung, L. *System Identification: Theory for the User* . Prentice-Hall, 1987.

[47] Astrom, K. and Wittenmark, B., *Adaptive Control*, Addison Wesley, 1995.

[48] Funahashi, K.-I. On the approximate realization of continuous mappings by neural networks. *Neural Networks*, 2 (3), 183-192, 1989.

[49] Hornik, K., Stinchcombe, M., and White, H. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2, 359-366, 1989.

[50] Stinchcombe, M. and White, H. Universal approximation using feedforward networks with non-sigmoidal hidden layer activation functions. In *Proc. of the International Joint Conference on Neural Networks*, pp. 607-611, Wash., D.C., 1989.

[51] Elliott, D. Reconstruction of nonlinear systems using delay lines and feedforward networks. Technical report, Neurodyne, Inc. and Institute for Systems Research, University of Maryland, 1995.

[52] Polycarpou, M. and Ioannou, P. Identification and control of nonlinear systems using neural network models: Design and stability analysis. Technical Report 91-09-01, University of Southern California, 1991.

[53] Hirsch, M. and Smale, S. *Differential Equations, Dynamical Systems, and Linear Algebra*. Academic Press, New York 1974.

[54] Hale, J. *Theory of Functional Differential Equations*. Springer-Verlag, New York, 1975.

[55] Seidl, D. and Lorenz, R. A structure by which a recurrent neural network can approximate a nonlinear dynamic system. In *Proceedings of the IJCNN*, 2, pp. 709-714, Seattle, Washington, 1991.

[56] Funahashi, K.-I. And Nakamura, Y. Approximation of dynamical systems by continuous-time recurrent neural networks. *Neural Networks*, 6, 801-806, 1993.

[57] Vidyasagar, M. *Nonlinear Systems Analysis*. Prentice Hall, New Jersey, 2nd edition, 1993.

[58] Sussman, H.J. Uniqueness of the weights for minimal feedforward nets with a given input-output map. *Neural Networks*, Vol. 5, pp. 589-593, 1992.

[59] Albertini, F. and Sontag, E.D. For neural networks functions determine form. *Neural Networks*, No. 6, pp. 975-990, 1992.

[60] Sontag, E.D. Neural Networks for control. in *Essays on control: Perspectives in the theory and its applications*, H.L. Trentelman and J.C. Willems (Efs.), Birkhauser, 1993.

[61] Slotine, J.-J. and Li, W. *Applied Nonlinear Control.* Prentice Hall, 1991.

[62] Sanner, R. and Slotine, J.-J. Direct adaptive control using Gaussian networks. *Proc. of ACC*, June 1991.

[63] Sanner, R. and Slotine, J.-J. Stable adaptive control and recursive identification using radial Gaussian networks. *Proc. of CDC*, pp. 203-205, Dec. 1991.

[64] Jaganathan, S. and Lewis, F.L. Identification of nonlinear dynamical systems using multilayered neural networks. *Automatica*, vol. 32, no. 12, pp. 1707-1712, 1996.

[65] Suykens, J., Vandewalle, J., and De Moor, B. Artificial Neural Networks for Modelling and Control of Non-Linear Systems. Kluwer Academic Publishers, Netherlands, 1996.

[66] Rugh, W. *Linear System Theory*. Prentice Hall, New Jersey, 1993.

[67] Isidori, A. *Nonlinear Control Systems.* Springer-Verlag, New York, 3rd edition, 1995.

[68] Sastry, S. and Bodson, M. *Adaptive Control*. Prentice-Hall, 1989.

[69] Luenberger, D. *Optimization by Vector Space Methods*. Wiley, 1969.

[70] Luenberger, D. *Linear and Nonlinear Programming*. Addison Wesley, 2nd edition, 1984.

[71] Werbos, P. *Beyond regression: New tools for predication and analysis in the behavioral sciences.* Ph.D. thesis, Harvard University, Cambridge, MA, 1974.

[72]Pineda, F. Generalization of back-propagation to recurrent neural networks. *Physical Review Letters*, 59 (19), 2229-2232, 1987.

[73] Werbos, P. Backpropagation through time: What it does and how to do it. In *Proceedings of the IEEE,* volume 78, pp. 1550-1560, 1990.

[74] Ioannou, P. and Datta, A. Robust adaptive control: Design, analysis and robustness bounds. In Robust Adaptive Control [Kokotovic, 1991].

[75] Praly, L., Bastin, G., and Jiang, Z. Adaptive stabilization of nonlinear systems. In *Adaptive Control* [Kokotivic91].