Identification and Control of Nonlinear Systems Using Neural Network Models: Design and Stability Analysis

by

Marios M. Polycarpou and Petros A. Ioannou

Report 91-09-01 **September 1991**

Identification and Control of Nonlinear Systems Using Neural Network Models: Design and Stability Analysis

Marios M. Polycarpou and **Petros A. Ioannou**

Department of Electrical Engineering-Systems
University of Southern California, MC-2563
Los Angeles, CA 90089-2563, U.S.A

email: polycarp@bode.usc.edu, ioannou@bode.usc.edu

Technical Report 91-09-01

September 1991

Abstract

The feasibility of applying neural network learning techniques in problems of system identification and control has been demonstrated through several empirical studies. These studies are based for the most part on gradient techniques for deriving parameter adjustment laws. While such schemes perform well in many cases, in general, problems arise in attempting to prove stability of the overall system, or convergence of the output error to zero. This paper presents a stability theory approach to synthesizing and analyzing identification and control schemes for nonlinear dynamical systems using neural network models. The nonlinearities of the dynamical system are assumed to be unknown and are modelled by neural network architectures. Multilayer networks with sigmoidal activation functions and radial basis function networks are the two types of neural network models that are considered. These static network architectures are combined with dynamical elements, in the form of stable filters, to construct a type of recurrent network configuration which is shown to be capable of approximating a large class of dynamical systems. Identification schemes based on neural network models are developed using two different techniques, namely, the Lyapunov synthesis approach and the gradient method. Both identification schemes are shown to guarantee stability, even in the presence of modelling errors. A novel network architecture, referred to as dynamic radial basis function network, is derived and shown to be useful in problems dealing with learning in dynamic enviroments. For a class of nonlinear systems, a stable neural network based control configuration is presented and analyzed.

1 Introduction

Adaptive identification and control of dynamical systems has been an active area of research for the last three decades. Although methods for controlling linear time invariant plants with unknown parameters had been pursued since the 1960's, it was not until the last decade that stable adaptive laws were established [1]-[4]. Recent advances in nonlinear control theory and, in particular, feedback linearization techniques [5, 6], have initiated activity aimed at developing adaptive control schemes for nonlinear plant models [7]-[10]. This area, which came to be known as *adaptive nonlinear control*, deals with systems where the uncertainty is due to unknown parameters which appear linearly with respect to known nonlinearities. Therefore, adaptive control research so far has been directed towards systems with a special class of parametric uncertainties.

The emergence of the neural network paradigm as a powerful tool for learning complex mappings from a set of examples has generated a great deal of excitement in using neural network models for identification and control of dynamical systems with unknown nonlinearities [11, 12]. Due to their approximation capabilities as well as their inherent adaptivity features, artificial neural networks present a potentially appealing alternative to modelling of nonlinear systems. Furthermore, from a practical perspective, the massive parallelism and fast adaptability of neural network implementations provide more incentives for further investigating the connectionist approach in problems involving dynamical systems with unknown nonlinearities.

The feasibility of applying neural network architectures in identification and control has been demonstrated, through simulations, by several studies, including [13]-[17]. The problem was originally formulated by Narendra et.al. [13], whose work instigated further research in this area. For the most part, these studies are based on first replacing unknown functions in the difference equation by static neural networks and then, according to a quadratic cost function, deriving update laws using optimization methods. A particular optimization procedure that has attracted a lot of attention is the steepest descent, or gradient method, which leads to the static backpropagation, or dynamic backpropagation-type algorithms [18]-[20], depending on whether the dynamical behavior of the system is taken into account or not. While such schemes perform well in many cases, in general, problems arise in attempting to prove stability of the overall system, or convergence of the output error to zero. Interestingly, methods based on straightforward application of optimization techniques (such as sensitivity models [21] and the M.I.T. rule [22]), that dominated the early adaptive linear control literature, exhibited similar stability problems. The fact that, even for linear systems, such methods can lead to instability was shown in [23, 24] (see also [25]).

In this paper we present a design procedure, based on stability theory, for modelling, identification and adaptive control of continuous-time nonlinear dynamical systems using neural network architectures. Particular emphasis is placed on the synthesis and stability analysis of the proposed schemes. The techniques developed here share some fundamental features with the parametric methods of both adaptive nonlinear control as well as adaptive linear control theory. Therefore, like earlier works [26, 13, 20, 27] this study also serves as an attempt to unify learning methods used by connectionists and adaptive control theorists. More specifically, one of the objectives of the paper is the development of stable neural network learning techniques for dynamic enviroments.

Our approach in this paper is based on *continuous-time* system representations. In particular, the plant dynamics are considered to evolve in continuous-time and furthermore, adaptive laws based on continuous adjustment of the weights are developed and analyzed. Due to the convenient "tapped delay line" representation, most of the current literature on identification and control of nonlinear systems using neural networks is developed in a discrete-time framework with iterative weight update rules. However, as also pointed out in [28], studying the problem in discrete-time has several drawbacks. It is known that discretization of physical continuous-time nonlinear systems yields highly complex discrete-time models [6]. More precisely, discretization of affine (in the input) continuous-time systems results in non-affine discrete-time models which are almost impossible to analyze. The approach usually taken by nonlinear control theorists is to design a control system based on continuous-time plant model and controller and then study the effect of sampling on the overall system [29]. Furthermore, from a practical perspective, if the neural network approach to control is to achieve its full potential, then it will be through parallel hardware realizations of neural network architectures implemented for real-time control systems. The current trend towards analog realizations of neural networks suggests considering identification and control schemes based on continuous-time systems with continuous-time adaptation.

From a theoretic point of view, neural networks can be considered as versatile mappings whose response to a specific input or pattern is determined by the values of adjustable weights. In our analysis we consider two types of neural network architectures: 1) multilayer neural network models with sigmoidal nonlinearities and 2) radial basis function networks with Gaussian activation functions. Multilayer neural networks [30] are by far the most widely used neural network models. They have been applied successfully to a wide range of problems including pattern and speech recognition, image compression, signal prediction and classification [31]. Due to some desirable features such as local adjustment of the weights and mathematical tractability, radial basis function networks have recently also attracted considerable attention especially in applications dealing with prediction and classification [32]-[34]. The importance of radial basis function networks has also greatly benefitted from the work of Poggio et.al. [35]-[37], where the relationship between regularization theory and radial basis function networks is explored.

The approximation capabilities of static sigmoidal type networks and of radial basis function networks has been studied by several research groups (see for example, [38]-[40]). In Section 2 we use these results to show that a proposed combination of static neural networks with dynamical components, such as stable filters, form a type of recurrent network capable of approximating a large class of dynamical systems. More precisely, it is shown that there exists a set of weights such that for a given input, the outputs of the real system and the proposed recurrent neural network model remain arbitrarily close over a finite interval of time.

System identification consists of first choosing an appropriate identification model and then adjusting the parameters of the model according to some adaptive law such that the response of the model to an input signal approximates the response of the real system to the same input. Since a mathematical characterization of a system is often a prerequisite to analysis and controller design, system identification is important not only for understanding and predicting the behavior of the process, but also for designing effective control laws. In Section 3 we develop and analyze two different neural network based identification schemes. The first scheme relies on the Lyapunov synthesis approach while the second is derived based on optimization methods. In this context, we introduce a novel architecture, referred to as *dynamic radial basis function* network, which is shown to be useful for applying optimization techniques in problems dealing with learning in dynamic enviroments. In Section 4, we consider the problem of controlling simple classes of nonlinear systems. Due to the intricacies of the problem, we first discuss in detail three types of mechanisms that can lead to instability, namely, the parameter drift, controllability and transient behavior problems, and propose modifications to the standard control and adaptive laws for dealing with these problems. Based on these modifications, we prove stability of the overall control system. Finally, in Section 5 we discuss the contribution of the paper and draw some final conclusions. Due to the length of the paper, simulation results are omitted.

2 Modelling of Dynamical Systems by Neural Networks

One of the basic requirements in using neural network[1] architectures to represent, identify and control nonlinear dynamical systems is the capability of these architectures to accurately model the behavior of a large class of dynamical systems that are encountered in science and engineering problems. This leads to the question of whether a given neural network configuration is able to approximate, in some appropriate sense, the input-output response of a class of dynamical systems. In order to be able to approximate the behavior of a dynamical system, it is clear that any proposed neural network configuration must have some feedback connections. In the neural network literature such networks are known as *recurrent* networks. The input-output response of neural networks, whether static or recurrent, is determined by the values of a set of parameters which are referred to as weights. Therefore the representational capabilities of a given network depend on whether there exists a set of weight values such that the neural network configuration approximates the behavior of a given dynamical system. The terms "weights" and "parameters" are used interchangeably throughout the paper.

In this section we consider the problem of constructing a neural network architecture that is capable of approximating the behavior of continuous-time dynamical systems, whose input-state-output representation is described by

$$\begin{aligned} \dot{x}(t) &= f(x(t), u(t)) \qquad\qquad x(0) = x^0 \\ y(t) &= h(x(t), u(t)) \end{aligned} \tag{2.1}$$

[1]The term "neural network" is used very loosely since some of the architectures considered in this paper have very little, if any, relation to biological neurons. In this context, "neural networks" are better interpreted as versatile mappings represented by the composition of many basic functions structured in a parallel fashion.

Figure 1: A recurrent neural network configuration for modelling the general dynamical system described by (2.1).

where $u \in \mathcal{R}^m$ is the input, $x \in \mathcal{R}^n$ is the state, $y \in \mathcal{R}^p$ is the output and $t \in \mathcal{R}^+$ is the temporal variable. The input u belongs to a class $\mathcal{U}$ of (piecewise continuous) admissible inputs. By adding and subtracting Ax, where A is a Hurwitz or stability matrix (i.e, has all of its eigenvalues in the open left-half complex plane), (2.1) becomes

$$\dot{x} = Ax + g(x,u)\,, \qquad\qquad y = h(x,u) \tag{2.2}$$

where $g(x,u) := f(x,u) - Ax$. Based on (2.2), we construct a recurrent network model by replacing the mappings g and h by feedforward (static) neural network architectures, denoted by N_1 and N_2 respectively. Therefore we consider the model

$$\begin{array}{rcll} \dot{\hat{x}} & = & A\hat{x} + \hat{g}(\hat{x}, u, \theta_g) & \qquad \hat{x}(0) = \hat{x}^0 \\ \hat{y} & = & \hat{h}(\hat{x}, u, \theta_h) & \end{array} \tag{2.3}$$

where $\hat{g}$ and $\hat{h}$ are the outputs of the static neural networks N_1 and N_2 respectively, while θ_g and θ_h denote the adjustable weights of these networks. In (2.3), $\hat{x}$ and $\hat{y}$ denote the state and output respectively of the recurrent network model.

Corresponding to the Hurwitz matrix A, we let $W(s) := (sI - A)^{-1}$ be an $n \times n$ matrix whose elements are stable transfer functions and s denotes the differential (Laplace) operator. Based on this definition of $W(s)$ as a stable filter, a block diagram representation of the recurrent network model described by (2.3) is depicted in Figure 1. This interconnection of static neural nets and dynamic components is proposed for modelling the input-output response of the general dynamical system described by (2.1). If we suppose that the real system and the proposed model are initially at the same state (i.e, $\hat{x}^0 = x^0$), then the natural question to ask is whether there exist weights θ_g^*, θ_h^* such that the input-output behavior $(u \mapsto \hat{y})$ of the neural network model (2.3) approximates, in some sense, the input-output behavior $(u \mapsto y)$ of the real system (2.1). This leads to the *validity* of the proposed model.

In examining this question, we will impose the following mild assumptions on the system to be approximated:

Figure 2: Block diagram representation of a two-layer sigmoidal neural network.

(S1) Given a class $\mathcal{U}$ of admissible inputs, then for any $u \in \mathcal{U}$ and any finite initial condition x^0 the state and output trajectories do not escape to infinity in finite time, i.e, for any finite $T > 0$ we have $|x(T)| + |y(T)| < \infty$.

(S2) The vector fields $f : \mathcal{R}^{n+m} \mapsto \mathcal{R}^n$ and $h : \mathcal{R}^{n+m} \mapsto \mathcal{R}^p$ are continuous with respect to their arguments. Furthermore, f satisfies a local Lipschitz condition so that the solution $x(t)$ to the differential equation (2.1) is unique for any finite initial condition x^0 and $u \in \mathcal{U}$.

The above assumptions are required in order to guarantee that the solution to the system described by (2.1) exists and is unique for any finite initial condition x^0 and any admissible input $u \in \mathcal{U}$.

We will also assume that the static neural network topologies N_i, $i = 1, 2$, that are used to represent the mappings g and h satisfy the following conditions:

(N1) Given a positive constant ε and a continuous function $f : \mathcal{C} \mapsto \mathcal{R}^p$, where $\mathcal{C} \subset \mathcal{R}^q$ is a compact set, there exists a weight vector $\theta = \theta^*$ such that the output $\hat{f}(X, \theta)$ of the neural network architecture N_i with n^* nodes (where n^* may depend on ε and f) satisfies

$$\max_{X \in \mathcal{C}} \left| \hat{f}(X, \theta^*) - f(X) \right| \leq \varepsilon$$

(N2) The output $\hat{f}(X, \theta)$ of the neural network architecture N_i is continuous with respect to its arguments for all finite (X, θ).

We next describe two popular neural network architectures that satisfy conditions (N1), (N2).
(a) Multilayer sigmoidal neural networks: Multilayer neural networks with sigmoidal type of nonlinearities are by far the most widely used neural network models [30]. From a theoretic point of view, multilayer neural networks may be considered as versatile maps whose response to a specific input is determined by the values of adjustable weights. The input-output behavior ($x \mapsto y$) of a two-layer sigmoidal neural network[2] (shown in Figure 2) with m inputs, n outputs and n^* hidden units or neurons is described (through the intermediate n^*-dimensional vectors $\bar{z}$, z) by

[2] In this paper we use the notational convention that a k-layer network consists of $k - 1$ hidden layers.

Figure 3: Block diagram representation of a Radial Basis Function neural network.

$$
\begin{aligned}
\bar{z} &= A_1 x + b \\
z_i &= \sigma(\bar{z}_i) \qquad\qquad i = 1,\ 2,\ \ldots\ n^* \\
y &= A_2 z
\end{aligned}
$$

where $x \in \mathcal{R}^m$ is the network input, $\bar{z} \in \mathcal{R}^{n^*}$ is the input to the sigmoidal nonlinearities, $z \in \mathcal{R}^{n^*}$ is the output of the hidden layer and $y \in \mathcal{R}^n$ is the network output; the adjustable weights of the network are the elements of $A_1 \in \mathcal{R}^{n^* \times m}$, $A_2 \in \mathcal{R}^{n \times n^*}$, $b \in \mathcal{R}^{n^*}$. The weight vector b is known as the *offset* or *bias* weight vector. In order to satisfy (N2), the sigmoidal nonlinearity $\sigma : \mathcal{R} \mapsto \mathcal{R}$ should be continuous. The hyperbolic tangent and the logistic function are examples of continuous sigmoids that are often used in neural network applications [31]. It is well known (see e.g, [38, 39, 41] and references therein) that if n^* is large enough then there exist weight values A_1, A_2, b such that the above two-layer neural network can approximate any continuous function $f(x)$ to any degree of accuracy over a compact set. This holds for any function σ that is nonconstant, bounded and nondecreasing. Therefore multilayer neural networks with as few as two layers satisfy (N1) and hence can be used to model continuous nonlinearities in dynamical systems. Furthermore, it has been shown [42] that three-layer neural networks (i.e, with two hidden layers), are capable of approximating discontinuous functions. Therefore, increasing the number of layers results in a natural enlargement of the class of functions that can be approximated.

(b) Radial basis function neural networks: Radial Basis Function (RBF) networks were introduced to the neural network literature by Broomhead et.al. [43] and have since gain significance in the field due to several application and theoretical results [32, 33, 35]. Recently, RBF networks have also been considered in adaptive control of nonlinear dynamical systems [28]. The input-output response ($x \mapsto y$) of a RBF neural network (shown in Figure 3) with m inputs, n outputs and n^* hidden, or kernel units, is characterized by

$$
\begin{aligned}
\xi_i &= g\left(|x - c_i|/\sigma_i\right) \qquad\qquad i = 1,\ 2,\ \ldots\ n^* \\
y &= A\xi
\end{aligned}
$$

where $x \in \mathcal{R}^m$ is the input, $\xi \in \mathcal{R}^{n^*}$ is the output of the hidden layer, $y \in \mathcal{R}^n$ is the output of the network; $A \in \mathcal{R}^{n \times n^*}$ is the weight matrix, while $c_i \in \mathcal{R}^m$ and $\sigma_i > 0$ are the center and width (or smoothing factor) of the ith kernel unit respectively. The Euclidean or a weighted

Euclidean norm $|\cdot|$ is often used. The continuous function $g : [0,\infty) \mapsto \mathcal{R}$ is the activation function which is usually chosen to be the Gaussian function $g(\beta) := e^{-\beta^2}$. Depending on the application, the centers c_i and/or widths σ_i of the network can either be adjustable (during learning) or they can be fixed. For example, in control applications it is very crucial for analytical purposes to choose c_i, σ_i apriori according to some preliminary training or an *ad-hoc* procedure and keep these values fixed during the learning phase. By doing so, the nonlinearities $g(\cdot)$ appear linearly with respect to the adjustable weights, which, as we will see later, simplifies the analysis considerably. It has recently been shown that under mild assumptions RBF neural networks are capable of universal approximation, i.e approximation of any continuous function over a compact set to any degree of accuracy [40, 44]. Therefore, Gaussian RBF networks also satisfy (N1), (N2) and hence they are candidates for modelling nonlinearities of dynamical systems.

Remark 2.1: In addition to sigmoidal and RBF neural networks, there are, of course, other representations that can be used for approximating static maps. Considerable more studies, both of theoretic and of practical nature, need to be performed before it is clear which architecture is best for approximating different classes of functions. Practical issues such as cost of hardware implementation, flexibility to increasing the number of nodes, speed of weight adjustment, as well as theoretic considerations such as number of parameters required in the approximation and robustness, are very important in choosing an appropriate network design. $\diamond$

Using Assumptions (S1)-(S2), (N1)-(N2), the following theorem establishes the capability of the proposed recurrent network architecture depicted in Figure 1 to approximate the behavior of the real system over a finite interval of time.

Theorem 1 *Suppose $x(0) = \hat{x}(0) = x^0$ and $u \in \mathcal{U} \subset \mathcal{R}^m$ where $\mathcal{U}$ is some compact set. Then given $\epsilon > 0$ and a finite $T > 0$, there exist weight values θ_g^*, θ_h^* such that for all $u \in \mathcal{U}$ the outputs of the real system and the recurrent neural network model satisfy*

$$\max_{t\in[0,T]} |y(t) - \hat{y}(t)| \;\leq\; \epsilon$$

The proof of Theorem 1 is given in Appendix A and relies on standard techniques from the theory of ordinary differential equations (see for example [45]).

Based on the above result, we will assume in the subsequent sections that the nonlinear dynamical system to be identified and/or controlled is represented by a recurrent network configuration with static neural networks replacing the unknown nonlinearities. In this framework, the real system is parametrized by neural network models with known underlying structure and unknown parameters or weights. In order to accomodate for modelling inaccuracies arising, for example, from having insufficient number of adjustable weights, we will allow the presence of modelling errors, which appear as additive disturbances in the differential equation representing the system model.

Figure 4: A general configuration for identification of nonlinear dynamical systems based on the series-parallel model.

3 Identification

In this section we consider the identification of nonlinear systems of the form

$$\dot{x} = f(x) + g(x)u \tag{3.1}$$

where $u \in \mathcal{R}$ is the input, $x \in \mathcal{R}^n$ is the state, which is assumed to be available for measurement, and f, g are smooth vector fields defined on an open set of $\mathcal{R}^n$. The above class of continuous-time nonlinear systems are called *affine systems* because in Equation (3.1) the control input u appears linearly with respect to g. There are several reasons for considering this class of nonlinear systems. First, most of the systems encountered in engineering are by nature (or design) affine systems. Secondly, most of the nonlinear control techniques, including feedback linearization, are developed for affine systems. Finally we note that non-affine systems (as described by (2.1)) can be converted to affine systems by passing the input through integrators [6]. This procedure is known as *dynamic extension*.

The problem of identification consists of choosing an appropriate identification model and adjusting the parameters of the model according to some adaptive law such that the response $\hat{x}$ of the model to an input signal u (or a class of input signals) approximates the response x of the real system to the same input. Since a mathematical characterization of a system is often a prerequisite to analysis and controller design, system identification is important not only for understanding and predicting the behavior of the system, but also for obtaining an effective control law. In this paper we consider identification schemes that are based on the setting shown in Figure 4, which is known as the *series-parallel configuration* [13].

As is common in identification procedures, we will assume that the state $x(t)$ is bounded for all admissible bounded inputs $u(t)$. Note that even though the real system is bounded-input bounded-state (BIBS) stable, there is no apriori guarantee that the output $\hat{x}$ of the identification model or that the adjustable parameters in the model will remain bounded.

Stability of the overall scheme depends on the particular identification model that is used as well as on the parameter adjustment rules that are chosen. This section is concerned with the development of identification models, based on sigmoidal and RBF neural networks, and the derivation of adaptive laws that guarantee stability of the overall identification structure.

Following the results of Section 2, the unknown nonlinearities $f(x)$ and $g(x)$ are parametrized by static neural networks with outputs $\hat{f}(x,\theta_f)$ and $\hat{g}(x,\theta_g)$ respectively, where $\theta_f \in \mathcal{R}^{n_f}$, $\theta_g \in \mathcal{R}^{n_g}$ are the adjustable weights and n_f, n_g denote the number of weights in the respective neural network approximation of f and g. By adding and subtracting the terms $\hat{f}$ and $\hat{g}u$, the nonlinear system described by (3.1) is rewritten as

$$\dot{x} = \hat{f}(x,\theta_f^*) + \hat{g}(x,\theta_g^*)u + \left[f(x) - \hat{f}(x,\theta_f^*)\right] + \left[g(x) - \hat{g}(x,\theta_g^*)\right] u \tag{3.2}$$

where θ_f^*, θ_g^* denote the optimal weight values (in the L_∞-norm sense) in the approximation of $f(x)$ and $g(x)$ respectively, for x belonging to a compact set $\mathcal{X} \subset \mathcal{R}^n$. For a given class of bounded input signals u, the set $\mathcal{X}$ is such that it contains all possible trajectories $x(t)$. While being aware of its existence, in our analysis we do not need to know the region $\mathcal{X}$.

Although the "optimal" weights θ_f^*, θ_g^* in (3.2) could take arbitrarily large values, from a practical perspective we are interested only in weights that belong to a (large) compact set. Therefore we will consider "optimal" weights θ_f^*, θ_g^* that belong to the convex compact sets $\mathcal{B}(M_f)$, $\mathcal{B}(M_g)$ respectively, where M_f, M_g are design constants and $\mathcal{B}(M) := \{\theta : |\theta| \leq M\}$ denotes a ball of radius M. In the adaptive law, the estimates of θ_f^*, θ_g^*, which are the adjustable weights in the approximation networks, are also restricted to $\mathcal{B}(M_f)$, $\mathcal{B}(M_g)$ respectively, through the use of a projection algorithm. By doing so, we avoid any numerical problems that may arise due to having weight values that are too large; furthermore, the projection algorithm prevents the weights from drifting to infinity, which, as will be apparent later, is a phenomenon that may occur with standard adaptive laws.

To summarize, the optimal weight vector θ_f^* is defined as the element in $\mathcal{B}(M_f)$ that minimizes $|f(x) - \hat{f}(x,\theta_f)|$ for $x \in \mathcal{X} \subset \mathcal{R}^n$; i.e,

$$\theta_f^* := \arg \min_{\theta_f \in \mathcal{B}(M_f)} \left\{ \sup_{x\in\mathcal{X}} \left| f(x) - \hat{f}(x,\theta_f)\right| \right\} \tag{3.3}$$

Similarly, θ_g^* is defined as

$$\theta_g^* := \arg \min_{\theta_g \in \mathcal{B}(M_g)} \left\{ \sup_{x\in\mathcal{X}} |g(x) - \hat{g}(x,\theta_g)| \right\} \tag{3.4}$$

Finally, it is noted that if the optimal weights are not unique then θ_f^* (and correspondingly θ_g^*) denotes an arbitrary (but fixed) element of the set of optimal weights.

Equation (3.2) is now expressed in compact form as

$$\dot{x} = \hat{f}(x,\theta_f^*) + \hat{g}(x,\theta_g^*)u + \nu(t) \tag{3.5}$$

where $\nu(t)$ denotes the *modelling error*, defined as

$$\nu(t) := \left[f(x(t)) - \hat{f}(x(t), \theta_f^*)\right] + \left[g(x(t)) - \hat{g}(x(t), \theta_g^*)\right] u(t)$$

The modelling error $\nu(t)$ is bounded by a constant ν_0 where

$$\nu_0 := \sup_{t \geq 0} \left| \left(f(x(t)) - \hat{f}(x(t), \theta_f^*)\right) + \left(g(x(t)) - \hat{g}(x(t), \theta_g^*)\right) u(t) \right|$$

Since by assumption, $u(t)$ and $x(t)$ are bounded, the constant ν_0 is finite. The value of ν_0 depends on many factors, such as the type of neural network that is used, the number of weights and layers, as well as the "size" of the compact sets $\mathcal{X}$, $\mathcal{B}(M_f)$, $\mathcal{B}(M_g)$. For example, the constraint that the optimal weights θ_f^*, θ_g^* belong to the sets $\mathcal{B}(M_f)$ and $\mathcal{B}(M_g)$ respectively, may increase the value of ν_0. However, if the constants M_f and M_g are large then any increase will be very small. In general, if the networks $\hat{f}$ and $\hat{g}$ are constructed appropriately then ν_0 will be a small number. Unfortunately, at present, the factors that influence how well a network is constructed, such as the number of weights and number of layers, are chosen for the most part by trial and error or other ad-hoc techniques. Therefore, a very attractive feature of our synthesis and analysis procedure is that we do not need to know the value of ν_0.

By replacing the unknown nonlinearities with feedforward neural network models, we have essentially rewritten the system (3.1) in the form (3.5), where the parameters θ_f^*, θ_g^* and the modelling error $\nu(t)$ are unknown, but the underlying structure of $\hat{f}$ and $\hat{g}$ is known. Based on (3.5), we next develop and analyze various types of identification schemes using both Gaussian RBF networks and multilayer network models with sigmoidal nonlinearities.

3.1 RBF Network Models

We first consider the case where the network architectures employed for modelling f and g are RBF networks. Therefore the functions $\hat{f}$ and $\hat{g}$ in (3.5) take the form

$$\hat{f} = W_1^* \xi(x), \qquad \hat{g} = W_2^* \zeta(x) \tag{3.6}$$

where W_1^*, W_2^* are $n \times n_1$ and $n \times n_2$ matrices respectively, representing in the spirit of (3.3), (3.4), the optimal weight values, subject to the constraints $\|W_1^*\|_F \leq M_1$, $\|W_2^*\|_F \leq M_2$. The norm $\|\cdot\|_F$ denotes the Frobenius matrix norm [46], defined as $\|A\|_F^2 := \sum_{ij} |a_{ij}|^2 = \mathrm{tr}\left\{AA^T\right\}$, where $\mathrm{tr}\{\cdot\}$ denotes the trace of a matrix. The constants n_1, n_2 are the number of kernel units in each approximation and the vector fields $\xi(x) \in \mathcal{R}^{n_1}$, $\zeta(x) \in \mathcal{R}^{n_2}$, which we refer to as *regressors*, are Gaussian type of functions, defined element-wise as

$$\begin{aligned} \xi_i(x) &= e^{-|x-c_{1i}|^2/\sigma_{1i}^2} \qquad & i = 1,\ 2,\ \cdots\ n_1 \\ \zeta_j(x) &= e^{-|x-c_{2j}|^2/\sigma_{2j}^2} \qquad & j = 1,\ 2,\ \cdots\ n_2 \end{aligned}$$

For analysis, it is crucial that the centers c_{1i}, c_{2j} and widths σ_{1i}, σ_{2j}, $i = 1, \cdots n_1$, $j = 1, \cdots n_2$, are chosen apriori. By doing so, the only adjustable weights are W_1, W_2, which appear linearly with respect to the nonlinearities ξ and ζ respectively. Based on "local tuning" training techniques several researchers have suggested methods for appropriately choosing the centers and widths of the radial basis functions [32, 47]. In this paper, we will simply assume that c_{1i}, c_{2j}, σ_{1i}, σ_{2j} are chosen apriori and kept fixed during adaptation of W_1, W_2.

Generally, in the problem of identification of nonlinear dynamical systems one is usually interested in obtaining an accurate model in a (possibly large) neighborhood $\mathcal{N}(x^0)$ of an equilibrium point $x = x^0$ and therefore it is intuitively evident that the centers should be clustered around x^0 in this neighborhood. Clearly, the number of kernel units and the position of the centers and widths will affect the approximation capability of the model and consequently the value of the modelling error $\nu(t)$. Hence, current and future research dealing with effectively choosing these quantities is also relevent to the topics discussed in this paper.

By substituting (3.6) in (3.5) we obtain

$$\dot{x} = W_1^* \xi(x) + W_2^* \zeta(x) u + \nu \tag{3.7}$$

Based on the RBF network model described by (3.7), we next develop parameter update laws for stable identification using various techniques derived from the Lyapunov synthesis approach and also basic optimization methods.

3.1.1 Lyapunov Synthesis Method

The RBF network model (3.7) is rewritten in the form

$$\dot{x} = -\alpha x + \alpha x + W_1^* \xi(x) + W_2^* \zeta(x) u + \nu \tag{3.8}$$

where $\alpha > 0$ is a scalar (design) constant. Based on (3.8) we consider the identification model

$$\dot{\hat{x}} = -\alpha \hat{x} + \alpha x + W_1 \xi(x) + W_2 \zeta(x) u \tag{3.9}$$

where W_1, W_2 are the estimates of W_1^*, W_2^* respectively, while $\hat{x}$ is the output of the identification model. The identification model (3.9), which we refer to as *RBF error filtering model*, is similar to estimation schemes developed in [9]. The RBF error filtering model is depicted in Figure 5. As can be seen from the figure, this identification model consists of two RBF network architectures in parallel and n first order stable filters $h(s) = 1/(s + \alpha)$.

If we define $e_x := \hat{x} - x$, the state error, and $\Phi_1 := W_1 - W_1^*$, $\Phi_2 := W_2 - W_2^*$, the weight estimation errors, then from (3.8), (3.9) we obtain the error equation

$$\dot{e}_x = -\alpha e_x + \Phi_1 \xi(x) + \Phi_2 \zeta(x) u - \nu \tag{3.10}$$

The Lyapunov synthesis method consists of choosing an appropriate Lyapunov function candidate V and selecting weight adaptive laws so that the time derivative $\dot{V}$ satisfies $\dot{V} \leq 0$. The Lyapunov method as a technique for deriving stable adaptive laws can be traced as far back as the 1960's, in the early literature of adaptive control theory for linear systems [23, 48].

Figure 5: A block diagram representation of the error filtering identification model developed using RBF networks.

In our case, an adaptive law for generating the parameter estimates $W_1(t)$, $W_2(t)$ is developed by considering the Lyapunov function candidate

$$V(e_x, \Phi_1, \Phi_2) \;=\; \frac{1}{2}|e_x|^2 + \frac{1}{2\gamma_1}\|\Phi_1\|_F^2 + \frac{1}{2\gamma_2}\|\Phi_2\|_F^2 \;=\; \frac{1}{2}e_x^T e_x + \frac{1}{2\gamma_1}\mathrm{tr}\left\{\Phi_1\Phi_1^T\right\} + \frac{1}{2\gamma_2}\mathrm{tr}\left\{\Phi_2\Phi_2^T\right\} \tag{3.11}$$

where γ_1, γ_2 are positive constants. These constants will appear in the adaptive laws and are referred to as *learning rates* or *adaptive gains*.

Using (3.10), the time derivative of V in (3.11) is expressed as

$$\dot{V} = -\alpha e_x^T e_x + \xi^T\Phi_1^T e_x + \frac{1}{\gamma_1}\mathrm{tr}\left\{\dot{\Phi}_1\Phi_1^T\right\} + \zeta^T\Phi_2^T e_x u + \frac{1}{\gamma_2}\mathrm{tr}\left\{\dot{\Phi}_2\Phi_2^T\right\} - e_x^T\nu$$

Using properties of the trace, such as

$$\xi^T\Phi_1^T e_x = \mathrm{tr}\left\{\xi^T\Phi_1^T e_x\right\} = \mathrm{tr}\left\{e_x\xi^T\Phi_1^T\right\}$$

we obtain

$$\begin{aligned}
\dot{V} &= -\alpha|e_x|^2 + \mathrm{tr}\left\{e_x\xi^T\Phi_1^T + \frac{1}{\gamma_1}\dot{\Phi}_1\Phi_1^T\right\} + \mathrm{tr}\left\{e_x u\zeta^T\Phi_2^T + \frac{1}{\gamma_2}\dot{\Phi}_2\Phi_2^T\right\} - e_x^T\nu \\
&= -\alpha|e_x|^2 + \frac{1}{\gamma_1}\mathrm{tr}\left\{\left(\gamma_1 e_x\xi^T + \dot{\Phi}_1\right)\Phi_1^T\right\} + \frac{1}{\gamma_2}\mathrm{tr}\left\{\left(\gamma_2 e_x u\zeta^T + \dot{\Phi}_2\right)\Phi_2^T\right\} - e_x^T\nu
\end{aligned} \tag{3.12}$$

Since W_1^*, W_2^* are constant, we have that $\dot{W}_1 = \dot{\Phi}_1$ and $\dot{W}_2 = \dot{\Phi}_2$. Therefore it is clear from (3.12) that if the parameter estimates W_1, W_2 are generated according to the adaptive laws

$$\dot{W}_1 = -\gamma_1 e_x\xi^T\,, \qquad \dot{W}_2 = -\gamma_2 e_x u\zeta^T \tag{3.13}$$

then (3.12) becomes

$$\dot{V} \;=\; -\alpha|e_x|^2 - e_x^T\nu \;\le\; -\alpha|e_x|^2 + \nu_0|e_x| \tag{3.14}$$

If there is no modelling error (i.e, $\nu_0 = 0$), then from (3.14) we have that $\dot{V}$ is negative semidefinite; hence stability of the overall identification scheme is guaranteed. However in the presence of modelling error, if $|e_x| < \nu_0/\alpha$ then it is possible that $\dot{V} > 0$, which implies that the weights $W_1(t)$, $W_2(t)$ may drift to infinity with time. This problem, which is referred to as *parameter drift* [49], is well known in the adaptive control literature. Parameter drift has also been encountered in empirical studies of neural network learning, where it is usually referred to as *weight saturation.*

In order to avoid parameter drift, $W_1(t)$, $W_2(t)$ are confined to the sets $\{W_1 : \|W_1\|_F \le M_1\}$, $\{W_2 : \|W_2\|_F \le M_2\}$ respectively, through the use of a projection algorithm [50, 54]. In particular, the standard adaptive laws described by (3.13) are modified to

$$\dot{W}_1 \;=\; \begin{cases} -\gamma_1 e_x\xi^T & \text{if } \{\|W_1\|_F < M_1\} \text{ or } \{\|W_1\|_F = M_1 \text{ and } e_x^T W_1\xi \ge 0\} \\ \mathcal{P}\left\{-\gamma_1 e_x\xi^T\right\} & \text{if } \{\|W_1\|_F = M_1 \text{ and } e_x^T W_1\xi < 0\} \end{cases} \tag{3.15}$$

$$\dot{W}_2 \;=\; \begin{cases} -\gamma_2 e_x u\zeta^T & \text{if } \{\|W_2\|_F < M_2\} \text{ or } \{\|W_2\|_F = M_2 \text{ and } e_x^T W_2\zeta u \ge 0\} \\ \mathcal{P}\left\{-\gamma_2 e_x u\zeta^T\right\} & \text{if } \{\|W_2\|_F = M_2 \text{ and } e_x^T W_2\zeta u < 0\} \end{cases} \tag{3.16}$$

where $\mathcal{P}\{\cdot\}$ denotes the projection onto the supporting hyperplane, defined as

$$\mathcal{P}\left\{-\gamma_1 e_x\xi^T\right\} \;:=\; -\gamma_1 e_x\xi^T + \gamma_1\frac{e_x^T W_1\xi}{\|W_1\|_F^2}W_1 \tag{3.17}$$

$$\mathcal{P}\left\{-\gamma_2 e_x u\zeta^T\right\} \;:=\; -\gamma_2 e_x u\zeta^T + \gamma_2\frac{e_x^T W_2\zeta u}{\|W_2\|_F^2}W_2 \tag{3.18}$$

Therefore, if the initial weights are chosen such that $\|W_1(0)\|_F \le M_1$, $\|W_2(0)\|_F \le M_2$ then we have $\|W_1(t)\|_F \le M_1$, $\|W_2(t)\|_F \le M_2$ for all $t \ge 0$. This can be readily established by noting that whenever $\|W_1\|_F = M_1$ (and correspondingly for $\|W_2\|_F = M_2$) then

$$\frac{d}{dt}\left\{\|W_1(t)\|_F^2 - M_1^2\right\} \le 0$$

which implies that the parameter estimate is directed towards the inside or the surface of the ball $\{W_1 : \|W_1\|_F \le M_1\}$. It is worth noting that the projection modification causes the adaptive law to be discontinuous. However, the trajectory behavior on the discontinuity hypersurface is "smooth" and hence existence of a solution, in the sense of Carathèodory [45], is assured. The issue of existence and uniqueness of solutions in adaptive systems is treated in detail in [51].

With the adaptive laws (3.15), (3.16), Equation (3.12) becomes

$$\begin{aligned} \dot{V} \;&=\; -\alpha|e_x|^2 - e_x^T\nu + I_1^*\,\mathrm{tr}\left\{\frac{e_x^T W_1\xi}{\|W_1\|_F^2}W_1\Phi_1^T\right\} + I_2^*\,\mathrm{tr}\left\{\frac{e_x^T W_2\zeta u}{\|W_2\|_F^2}W_2\Phi_2^T\right\} \\ &\le\; -\alpha|e_x|^2 - e_x^T\nu + I_1^*\,\frac{e_x^T W_1\xi}{\|W_1\|_F^2}\,\mathrm{tr}\left\{W_1\Phi_1^T\right\} + I_2^*\,\frac{e_x^T W_2\zeta u}{\|W_2\|_F^2}\,\mathrm{tr}\left\{W_2\Phi_2^T\right\} \end{aligned} \tag{3.19}$$

where I_1^*, I_2^* are indicator functions defined as $I_1^* = 1$ if the conditions $\|W_1\|_F = M_1$ and $e_x^T W_1 \xi < 0$ are satisfied and $I_1^* = 0$ otherwise (and correspondingly for I_2^*). The following lemma establishes that the additional terms introduced by the projection can only make $\dot{V}$ more negative, which implies that the projection modification guarantees boundedness of the weights without affecting the rest of the stability properties established in the absence of projection.

Lemma 1 *Based on the adaptive laws (3.15), (3.16) the following inequalities hold:*

(i) $I_1^* \left(e_x^T W_1 \xi / \|W_1\|_F^2 \right) \; tr\left\{ W_1 \Phi_1^T \right\} \leq 0$.

(ii) $I_2^* \left(e_x^T W_2 \zeta u / \|W_2\|_F^2 \right) \; tr\left\{ W_2 \Phi_2^T \right\} \leq 0$.

The proof of Lemma 1 is given in Appendix B. Now, using Lemma 1, (3.19) becomes

$$\dot{V} \;\leq\; -\alpha |e_x|^2 - e_x^T \nu \;\leq\; -\alpha |e_x|^2 + \nu_0 |e_x| \tag{3.20}$$

Based on (3.20), we next summarize the properties of the weight adaptive laws (3.15), (3.16). It is pointed out that the proof of the following theorem employs well known techniques from the adaptive control literature. In the sequel, the notation $z \in L_2$ means $\int_0^\infty |z(t)|^2 dt < \infty$ while $z \in L_\infty$ implies $\sup_{t \geq 0} |z(t)| < \infty$.

Theorem 2 *Consider the error filtering identification scheme (3.9). The weight adaptive laws given by (3.15), (3.16) guarantee the following properties:*

(a) *For $\nu_0 = 0$ (no modelling error), we have*

- $e_x, \hat{x}, \Phi_1, \Phi_2 \in L_\infty, \quad e_x \in L_2$.
- $\lim_{t\to\infty} e_x(t) = 0, \quad \lim_{t\to\infty} \dot{\Phi}_1(t) = 0, \quad \lim_{t\to\infty} \dot{\Phi}_2(t) = 0$.

(b) *For $\sup_{t\geq 0} |\nu(t)| \leq \nu_0$ we have*

- $e_x, \hat{x}, \Phi_1, \Phi_2 \in L_\infty$.
- *there exist constants k_1, k_2 such that*

$$\int_0^t |e_x(\tau)|^2 d\tau \;\leq\; k_1 + k_2 \int_0^t |\nu(\tau)|^2 d\tau$$

Proof:
(a) With $\nu_0 = 0$, Equation (3.20) becomes

$$\dot{V} \;\leq\; -\alpha |e_x|^2 \;\leq\; 0 \tag{3.21}$$

Hence $V \in L_\infty$, which from (3.11) implies $e_x, \Phi_1, \Phi_2 \in L_\infty$. Furthermore, $\hat{x} = e_x + x$ is also bounded. Since V is a non-increasing function of time and bounded from below the $\lim_{t\to\infty} V(t) = V_\infty$ exists. Therefore by integrating (3.21) from 0 to ∞ we have

$$\int_0^\infty |e_x(\tau)|^2 d\tau \;\leq\; \frac{1}{\alpha}\left[V(0) - V_\infty\right] \;<\; \infty$$

which implies that $e_x \in L_2$. By the definition of the Gaussian radial basis function, the regressor vectors $\xi(x)$ and $\zeta(x)$ are bounded for all x and by assumption u is also bounded. Hence from (3.10) we have that $\dot{e}_x \in L_\infty$. Since $e_x \in L_2 \cap L_\infty$ and $\dot{e}_x \in L_\infty$, using Barbalat's Lemma [3] we conclude that $\lim_{t\to\infty} e_x(t) = 0$. Now, using the boundedness of $\xi(t)$ and the convergence of $e_x(t)$ to zero, we have that $\dot{\Phi}_1 = \dot{W}_1$ also converges to zero. Similarly, $\dot{\Phi}_2 \to 0$ as $t \to \infty$.

(b) With the projection algorithm, it is guaranteed that $\|W_1\|_F \le M_1$, $\|W_2\|_F \le M_2$. Therefore the weight estimation errors are also bounded, i.e $\Phi_1, \Phi_2 \in L_\infty$. From (3.20) it is clear that if $|e_x| > \nu_0/\alpha$ then $\dot{V} < 0$ which implies that $e_x \in L_\infty$ and consequently $\hat{x} \in L_\infty$. In order to prove the second part, we proceed to complete the square in (3.20) :

$$\begin{aligned} \dot{V} &\le -\frac{\alpha}{2}|e_x|^2 - \frac{\alpha}{2}\left[|e_x|^2 + \frac{2}{\alpha}e_x^T\nu\right] \\ &\le -\frac{\alpha}{2}|e_x|^2 + \frac{1}{2\alpha}|\nu|^2 \end{aligned}$$

Therefore, by integrating both sides and using the fact that $V \in L_\infty$ we obtain

$$\begin{aligned} \int_0^t |e_x(\tau)|^2 d\tau &\le \frac{2}{\alpha}[V(0) - V(t)] + \frac{1}{\alpha^2}\int_0^t |\nu(\tau)|^2 d\tau \\ &\le k_1 + k_2 \int_0^t |\nu(\tau)|^2 d\tau \end{aligned}$$

where $k_1 := 2/\alpha \left(V(0) - \sup_{t\ge0} V(t)\right)$ and $k_2 := 1/\alpha^2$. □

Remark 3.1: For notational simplicity the above identification scheme was developed with the filter pole α and the learning rates γ_1, γ_2 being scalars. It can be easily verified that the analysis is still valid if $-\alpha$ in (3.9) is replaced by a Hurwitz matrix $A \in \mathcal{R}^{n\times n}$ and γ_1, γ_2 in the parameter update laws are replaced by positive definite learning rate matrices $\Gamma_1 \in \mathcal{R}^{n_1\times n_1}$, $\Gamma_2 \in \mathcal{R}^{n_2\times n_2}$ respectively. ◇

Remark 3.2: To ensure robustness of the identification scheme with respect to modelling errors, we have considered a projection algorithm, which prevents the weights from drifting to infinity. The stability of the proposed identification scheme in the presence of modelling errors can also be achieved by other modifications to the standard adaptive laws, such as the *fixed*, or *switching* σ*-modification* [49, 52], ε*-modification* [53] and the *dead-zone* [2, 3]. A comprehensive exposition to robust adaptive control theory for linear systems is given in [54]. ◇

Remark 3.3: Under the assumptions of Theorem 2 we cannot conclude anything about the convergence of the weights to their optimal values. In order to guarantee convergence, $\xi(x)$, $\zeta(x)u$ need to satisfy a *persistency of excitation* condition. A signal $z(t) \in \mathcal{R}^n$ is persistently exciting in $\mathcal{R}^n$ if there exist positive constants α_0, α_1, T such that

$$\alpha_0 I \le \int_t^{t+T} z(\tau)z^T(\tau)d\tau \le \alpha_1 I \qquad \forall t \ge 0$$

In contrast to linear systems where the persistency of excitation condition has been transformed into a condition on the input signal [2, 3], in nonlinear systems this condition cannot be verified apriori because the regressors are nonlinear functions of the state x. ◇

3.1.2 Optimization Methods

In order to derive optimization-based weight adjustment rules that also guarantee stability of the overall scheme, we need to develop an identification model in which the output error e_x is related to the weight estimation errors Φ_1, Φ_2 in a simple algebraic fashion. To achieve this objective we consider filtered forms of the regressors ξ, ζ and the state x. We start by rewriting (3.8) in the filter form

$$x = \frac{\alpha}{s+\alpha}[x] + W_1^* \frac{1}{s+\alpha}[\xi(x)] + W_2^* \frac{1}{s+\alpha}[\zeta(x)u] + \frac{1}{s+\alpha}[\nu] \tag{3.22}$$

where s denotes the differential (Laplace) operator.[3] As in Section 2, the notation $h(s)[z]$ is to be interpreted as the output of the filter $h(s)$ with z as the input. Equation (3.22) is now expressed in the compact form

$$x = x_f + W_1^* \xi_f + W_2^* \zeta_f + \nu_f \tag{3.23}$$

where x_f, ξ_f, ζ_f are generated by filtering x, ξ, ζu respectively:

$$\begin{array}{rcll} \dot{x}_f &=& -\alpha x_f + \alpha x & \qquad x_f(0) = 0 \\ \dot{\xi}_f &=& -\alpha \xi_f + \xi & \qquad \xi_f(0) = 0 \\ \dot{\zeta}_f &=& -\alpha \zeta_f + \zeta u & \qquad \zeta_f(0) = 0 \end{array}$$

Since $\nu(t)$ is bounded by ν_0, the filtered modelling error $\nu_f := \frac{1}{s+\alpha}[\nu]$ is another bounded disturbance signal, i.e $|\nu_f(t)| \leq \nu_0/\alpha$.

Based on (3.23), we consider the identification model

$$\hat{x} = x_f + W_1 \xi_f + W_2 \zeta_f \tag{3.24}$$

This model, which will be referred to as the *regressor filtering* identification model [9], is shown in Figure 6, in block diagram representation. The regressor filtering scheme requires $n+n_1+n_2$ first order filters, which is considerably more than the n filters required in the error filtering scheme. As can be seen from Figure 6, the filters appear inside the RBF networks, forming a *dynamic RBF network*. This neural network architecture can be directly applied to modelling of dynamical systems in the same way that RBF networks are used as models of static mappings.

Using the regressor filtering scheme, the output error $e_x := \hat{x} - x$ satisfies

$$e_x \;=\; W_1 \xi_f + W_2 \zeta_f + x_f - x \;=\; \Phi_1 \xi_f + \Phi_2 \zeta_f - \nu_f \tag{3.25}$$

where $\Phi_1 = W_1 - W_1^*$, $\Phi_2 = W_2 - W_2^*$ are the weight estimation errors. Hence by filtering the regressors, we obtain an algebraic relationship between the output error e_x and the weight

[3] In deriving (3.22) we have assumed (without loss of generality) zero initial condition for the state, i.e $x^0 = 0$. Note that if $x^0 \neq 0$ then the initial condition will appear in the identification model so that it gets cancelled in the error equation. This is possible because x is available for measurement.

Figure 6: A block diagram representation of the regressor filtering identification model developed using RBF networks.

estimation errors Φ_1, Φ_2. In the framework of the optimization approach [55], adaptive laws for W_1, W_2 are obtained by minimizing an appropriate cost functional with respect to each element of W_1, W_2. Here we consider an *instantaneous* cost functional with a constraint on the possible values that W_1, W_2 can take. This leads to the following constrained minimization problem:

$$\begin{array}{ll} \text{minimize} & \mathcal{J}(W_1, W_2) = \frac{1}{2} e_x^T e_x \\ \text{subject to} & \|W_1\|_F \leq M_1 \\ & \|W_2\|_F \leq M_2 \end{array} \tag{3.26}$$

Using the *gradient projection method* [55], we obtain the following adaptive laws for continuous adjustment of the weights W_1, W_2 :

$$\dot{W}_1 = \begin{cases} -\gamma_1 e_x \xi_f^T & \text{if } \{\|W_1\|_F < M_1\} \text{ or } \{\|W_1\|_F = M_1 \text{ and } e_x^T W_1 \xi_f \geq 0\} \\ \mathcal{P}\left\{-\gamma_1 e_x \xi_f^T\right\} & \text{if } \{\|W_1\|_F = M_1 \text{ and } e_x^T W_1 \xi_f < 0\} \end{cases} \tag{3.27}$$

$$\dot{W}_2 = \begin{cases} -\gamma_2 e_x \zeta_f^T & \text{if } \{\|W_2\|_F < M_2\} \text{ or } \{\|W_2\|_F = M_2 \text{ and } e_x^T W_2 \zeta_f \geq 0\} \\ \mathcal{P}\left\{-\gamma_2 e_x \zeta_f^T\right\} & \text{if } \{\|W_2\|_F = M_2 \text{ and } e_x^T W_2 \zeta_f < 0\} \end{cases} \tag{3.28}$$

where $\mathcal{P}\{\cdot\}$ denotes the projection operation defined in Section 3.1.1. The weight adaptive laws (3.27), (3.28) have the same form as (3.15), (3.16), which were derived by the Lyapunov method, with the exception that the output error e_x and the regressor vectors ξ, ζ are defined in a different way.

The counterpart of Theorem 2 concerning the stability properties of the regressor filtering identification scheme with the adaptive laws (3.27), (3.28), obtained using the gradient projection method is described by the following result.

Theorem 3 *Consider the regressor filtering identification scheme described by (3.24). The weight adaptive laws given by (3.27), (3.28) guarantee the following properties:*

(a) *For $\nu(t) = 0$ (no modelling error), we have*

- $e_x, \hat{x}, \Phi_1, \Phi_2 \in L_\infty, \quad e_x \in L_2$.
- $\lim_{t\to\infty} e_x(t) = 0, \quad \lim_{t\to\infty} \dot{\Phi}_1(t) = 0, \quad \lim_{t\to\infty} \dot{\Phi}_2(t) = 0$.

(b) *For $\sup_{t\geq 0} |\nu(t)| \leq \nu_0$, we have*

- $e_x, \hat{x}, \Phi_1, \Phi_2 \in L_\infty$.
- *there exist constants k_1, k_2 such that*

$$\int_0^t |e_x(\tau)|^2 d\tau \;\leq\; k_1 + k_2 \int_0^t |\nu(\tau)|^2 d\tau$$

Proof: Consider the Lyapunov function candidate

$$V(\Phi_1, \Phi_2) \;=\; \frac{1}{2\gamma_1}\|\Phi_1\|_F^2 + \frac{1}{2\gamma_2}\|\Phi_2\|_F^2 \;=\; \mathrm{tr}\left\{\frac{1}{2\gamma_1}\Phi_1\Phi_1^T + \frac{1}{2\gamma_2}\Phi_2\Phi_2^T\right\} \tag{3.29}$$

Using (3.25), Lemma 1 and the fact that $\dot{\Phi}_1 = \dot{W}_1$, $\dot{\Phi}_2 = \dot{W}_2$, the time derivative of V along (3.27), (3.28) can be expressed as

$$\begin{aligned}
\dot{V} \;&=\; \mathrm{tr}\left\{-e_x\xi_f^T\Phi_1^T - e_x\zeta_f^T\Phi_2^T\right\} + I_1^*\,\frac{e_x^T W_1\xi_f}{\|W_1\|_F^2}\,\mathrm{tr}\left\{W_1\Phi_1^T\right\} + I_2^*\,\frac{e_x^T W_2\zeta_f}{\|W_2\|_F^2}\,\mathrm{tr}\left\{W_2\Phi_2^T\right\} \\
&\leq\; -\mathrm{tr}\left\{e_x\left(\xi_f^T\Phi_1^T + \zeta_f^T\Phi_2^T\right)\right\} \\
&=\; -\mathrm{tr}\left\{e_x\left(e_x^T + \nu_f^T\right)\right\} \;=\; -|e_x|^2 - \nu_f^T e_x \;\leq\; -|e_x|^2 + \frac{\nu_0}{\alpha}|e_x|
\end{aligned} \tag{3.30}$$

(a) If $\nu(t) = 0$ then

$$\dot{V} \leq -|e_x|^2 \leq 0 \tag{3.31}$$

Therefore $V \in L_\infty$, which from (3.29) implies that $\Phi_1, \Phi_2 \in L_\infty$. Using this, together with the boundedness of ξ_f, ζ_f in (3.25) gives $e_x, \hat{x} \in L_\infty$. Furthermore, by integrating both sides of (3.31) from 0 to ∞ it can be shown that $e_x \in L_2$. Now by taking the time derivative of e_x in (3.25) we obtain

$$\dot{e}_x = \dot{\Phi}_1\xi_f + \Phi_1\dot{\xi}_f + \dot{\Phi}_2\zeta_f + \Phi_2\dot{\zeta}_f \tag{3.32}$$

Since $\dot{\Phi}_1, \xi_f, \Phi_1, \dot{\xi}_f, \dot{\Phi}_2, \zeta_f, \Phi_2, \dot{\zeta}_f \in L_\infty$, (3.32) implies that $\dot{e}_x \in L_\infty$ and thus using Barbalat's Lemma we conclude that $\lim_{t\to\infty} e_x(t) = 0$. Using the boundedness of ξ_f and ζ_f, it can be readily verified that $\dot{\Phi}_1$ and $\dot{\Phi}_2$ also converge to zero.
(b) Suppose $\sup_{t\geq 0}|\nu(t)| \leq \nu_0$. The projection algorithm guarantees that W_1, W_2 are bounded, which implies $\Phi_1, \Phi_2 \in L_\infty$. Since ξ_f, ζ_f are also bounded, from (3.25) we obtain $e_x \in L_\infty$ and also $\hat{x} \in L_\infty$. The proof of the second part follows directly along the same lines as its counterpart in Theorem 2. □

Remark 3.4: As in the Lyapunov method, the scalar learning rates γ_1, γ_2 in the parameter update laws (3.27), (3.28) can be replaced by positive definite matrices Γ_1, Γ_2. $\diamond$

Remark 3.5: Minimization of an appropriate *integral cost* using Newton's method yields the recursive least-squares algorithm. In the least-squares algorithm the learning rate matrices Γ_1, Γ_2 are time varying and are adjusted concurrently with the weights. Unfortunately, the least-squares algorithm is computationally very expensive and especially in neural network modelling, where the number of units n_1, n_2 are usually large, updating the matrices Γ_1, Γ_2, which consist of n_1^2, n_2^2 entries respectively, makes this algorithm impractical. $\diamond$

3.2 Multilayer Network Models

In this section we consider the case where the network structures employed in the approximation of $f(x)$ and $g(x)$ are multilayer neural networks with sigmoidal-type of activation functions. Although this is the most commonly used class of neural network models in empirical studies, there are very few analytical results concerning the stability properties of such networks in problems dealing with learning in dynamic enviroments. The main difficulty in analyzing the behavior of recurrent network architectures with feedforward multilayer neural networks as subsystems arises due to the fact that the adjustable weights appear non-affinely with respect to the nonlinearities of the network structure.

Our approach relies on developing an error identification scheme, based on the same procedure as in Section 3.1.1. The analysis proceeds through the use of a Taylor series expansion around the optimal weights. The adaptive law is designed based on the first order (linear) approximation of the Taylor series expansion. In this framework, the RBF error filtering scheme presented earlier, constitutes a special case of the analysis in this section, with the higher order terms not present and the regressor being independent of the weight values. As a consequence of the presence of higher order terms, the results obtained here are weaker, in the sense that, even if there is no modelling error it cannot be guaranteed that the output error will converge to zero. The significance of this analysis is based on proving that all the signals in the proposed identification scheme remain bounded, as well as developing a unified approach to synthesizing and analyzing stable dynamic learning configurations using different types of neural network architectures.

Consider the system (3.5), where θ_f^*, θ_g^* are the optimal weights in the minimization of $|f(x) - \hat{f}(x, \theta_f^*)|$ and $|g(x) - \hat{g}(x, \theta_g^*)|$ respectively, for $x \in \mathcal{X}$ and subject to the constraints $|\theta_f^*| \leq M_f$, $|\theta_g^*| \leq M_g$, where M_f, M_g are (large) design constants. The functions $\hat{f}$ and $\hat{g}$ are the outputs of multilayer neural networks with sigmoidal nonlinearities in-between layers. We start by adding and subtracting αx in (3.5), where α is a positive design constant. This gives

$$\dot{x} = -\alpha x + \alpha x + \hat{f}(x, \theta_f^*) + \hat{g}(x, \theta_g^*)u + \nu \tag{3.33}$$

Based on (3.33) we consider the error filtering identification model

$$\dot{\hat{x}} = -\alpha \hat{x} + \alpha x + \hat{f}(x, \theta_f) + \hat{g}(x, \theta_g)u \tag{3.34}$$

Figure 7: A block diagram representation of the error filtering identification model developed using multilayer sigmoidal neural networks.

where $\theta_f \in \mathcal{R}^{n_f}$, $\theta_g \in \mathcal{R}^{n_g}$ are the estimates of the optimal weights θ_f^*, θ_g^* respectively. The constants n_f, n_g are the number of weights in each approximation. The error filtering identification scheme described by (3.34) is shown in block diagram representation in Figure 7. From (3.33), (3.34), the output error $e_x = \hat{x} - x$ satisfies the differential equation:

$$\dot{e}_x = -\alpha e_x + \left[\hat{f}(x,\theta_f) - \hat{f}(x,\theta_f^*)\right] + \left[\hat{g}(x,\theta_g) - \hat{g}(x,\theta_g^*)\right] u - \nu \tag{3.35}$$

In order to obtain an adaptive law for the weights θ_f it is convenient to consider the first order approximation of the difference $\hat{f}(x,\theta_f) - \hat{f}(x,\theta_f^*)$. Using the Taylor series expansion[4] of $\hat{f}(x,\theta_f)$ around the point (x,θ_f^*) we obtain

$$\hat{f}(x,\theta_f) - \hat{f}(x,\theta_f^*) = \frac{\partial \hat{f}}{\partial \theta_f}(x,\theta_f) \cdot \left(\theta_f - \theta_f^*\right) + \hat{f}_0(x,\theta_f) \tag{3.36}$$

where $\hat{f}_0(x,\theta_f)$ represents the higher order terms (with respect to θ_f) of the expansion. If we define the weight estimation error as $\phi_f := \theta_f - \theta_f^*$ then from (3.36) we have that

$$\hat{f}_0(x,\theta_f) = \hat{f}(x,\theta_f) - \hat{f}(x,\theta_f^*) - \frac{\partial \hat{f}}{\partial \theta_f}(x,\theta_f) \cdot \phi_f$$

[4]Throughout the analysis we require that the network outputs $\hat{f}(x,\theta_f)$, $\hat{g}(x,\theta_g)$ are smooth functions of their arguments. This can easily be achieved if the sigmoid used is a smooth function. The logistic function and the hyperbolic tangent are examples of popular sigmoids that also satisfy the smoothness condition.

By the Mean Value Theorem there exists $\overline{\theta}_f \in [\theta_f,\ \theta_f^*]$, i.e, $\overline{\theta}_f = \lambda\theta_f + (1-\lambda)\theta_f^*$ for some $\lambda \in [0,\ 1]$, such that

$$\begin{aligned} \hat{f}_0(x,\theta_f) &= \frac{\partial \hat{f}}{\partial \theta_f}(x,\overline{\theta}_f)\cdot\phi_f - \frac{\partial \hat{f}}{\partial \theta_f}(x,\theta_f)\cdot\phi_f \\ &= \left[\frac{\partial \hat{f}}{\partial \theta_f}(x,\overline{\theta}_f) - \frac{\partial \hat{f}}{\partial \theta_f}(x,\theta_f)\right]\phi_f \end{aligned}$$

Now let

$$\delta_f := \sup_{x\in\mathcal{X},\ \theta_f\in\mathcal{B}(M_f),\ \overline{\theta}_f\in[\theta_f,\ \theta_f^*]} \left\| \frac{\partial \hat{f}}{\partial \theta_f}(x,\overline{\theta}_f) - \frac{\partial \hat{f}}{\partial \theta_f}(x,\theta_f)\right\| \tag{3.37}$$

where $\mathcal{B}(M_f)$ denotes a ball of radius M_f. Therefore, the higher order terms satisfy

$$|\hat{f}_0(x,\theta_f)| \ \leq\ \delta_f|\phi_f| \qquad \forall x\in\mathcal{X},\ \ \forall\theta_f\in\mathcal{B}(M_f) \tag{3.38}$$

Using the same procedure we have

$$\hat{g}(x,\theta_g) - \hat{g}(x,\theta_g^*) = \frac{\partial \hat{g}}{\partial \theta_g}(x,\theta_g)\cdot\phi_g + \hat{g}_0(x,\theta_g) \tag{3.39}$$

where $\phi_g := \theta_g - \theta_g^*$ and $\hat{g}_0(x,\theta_g)$ satisfies

$$|\hat{g}_0(x,\theta_g)| \ \leq\ \delta_g|\phi_g| \qquad \forall x\in\mathcal{X},\ \ \forall\phi_g\in\mathcal{B}(M_g) \tag{3.40}$$

and the constant δ_g is defined as

$$\delta_g := \sup_{x\in\mathcal{X},\ \theta_g\in\mathcal{B}(M_g),\ \overline{\theta}_g\in[\theta_g,\ \theta_g^*]} \left\| \frac{\partial \hat{g}}{\partial \theta_g}(x,\overline{\theta}_g) - \frac{\partial \hat{g}}{\partial \theta_g}(x,\theta_g)\right\| \tag{3.41}$$

From now on, for notational simplicity we define the *regressors*

$$\xi(x,\theta_f) := \frac{\partial \hat{f}}{\partial \theta_f}(x,\theta_f), \qquad \zeta(x,\theta_g) := \frac{\partial \hat{g}}{\partial \theta_g}(x,\theta_g)$$

where ξ, ζ are $n\times n_f$ and $n\times n_g$ matrices respectively, representing the sensitivity functions between the output of the network and the adjustable weights. It is worth noting that these sensitivity functions are exactly the functions that are calculated by the standard (static) backpropagation algorithm.

Now, by replacing (3.36) and (3.39) in (3.35) we obtain

$$\dot{e}_x = -\alpha e_x + \xi(x,\theta_f)\phi_f + \zeta(x,\theta_g)\phi_g u + \hat{f}_0(x,\theta_f) + \hat{g}_0(x,\theta_g)u - \nu \tag{3.42}$$

Once expressed in the form (3.42), it is clear that this error equation is a generalization of the corresponding error equation for RBF networks. In particular, if the higher order terms $\hat{f}_0$, $\hat{g}_0$ are identically zero and the regressors ξ, ζ are independent of θ_f and θ_g respectively, then (3.42) becomes of the same form as the error equation for RBF networks. In deriving adaptive laws for adjusting the weights θ_f and θ_g, we neglect the higher order terms $\hat{f}_0$ and $\hat{g}_0$. This is consistent with linearization techniques, whose main principle is that for ϕ_f, ϕ_g sufficiently small, the linearized terms $\xi\phi_f$ and $\zeta\phi_g$ dominate the higher order terms $\hat{f}_0$, $\hat{g}_0$. Based on these observations we consider the following adaptive laws for adjusting the weights θ_f, θ_g :

$$\dot{\theta}_f = \begin{cases} -\gamma_f \xi^T e_x & \text{if } \{|\theta_f| < M_f\} \text{ or } \{|\theta_f| = M_f\} \text{ and } e_x^T \xi \theta_f \geq 0\} \\ \mathcal{P}\left\{-\gamma_f \xi^T e_x\right\} & \text{if } \{|\theta_f| = M_f\} \text{ and } e_x^T \xi \theta_f < 0\} \end{cases} \tag{3.43}$$

$$\dot{\theta}_g = \begin{cases} -\gamma_g \zeta^T e_x u & \text{if } \{|\theta_g| < M_g\} \text{ or } \{|\theta_g| = M_g\} \text{ and } e_x^T \zeta \theta_g u \geq 0\} \\ \mathcal{P}\left\{-\gamma_g \zeta^T e_x u\right\} & \text{if } \{|\theta_g| = M_g\} \text{ and } e_x^T \zeta \theta_g u < 0\} \end{cases} \tag{3.44}$$

where $\mathcal{P}\{\cdot\}$ again denotes the projection operation, which is computed as follows:

$$\begin{aligned} \mathcal{P}\left\{-\gamma_f \xi^T e_x\right\} &:= -\gamma_f \xi^T e_x + \gamma_f \frac{e_x^T \xi \theta_f}{|\theta_f|^2} \theta_f \\ \mathcal{P}\left\{-\gamma_g \zeta^T e_x u\right\} &:= -\gamma_g \zeta^T e_x u + \gamma_g \frac{e_x^T \zeta \theta_g u}{|\theta_g|^2} \theta_g \end{aligned}$$

The weight adjustment laws (3.43), (3.44) are the usual adaptive laws obtained by the Lyapunov synthesis method, with the projection algorithm modification for preventing the weights from drifting to infinity.

Remark 3.6: An intuitive interpretation of the above adaptive laws based on more familiar optimization techniques can be obtained as follows: by setting $\dot{e}_x = 0$ in the error equation (3.35) and solving for the quasi-steady-state response e_{ss} of e_x we have

$$e_{ss} = \frac{1}{\alpha}\left[\left(\hat{f}(x, \theta_f) - \hat{f}(x, \theta_f^*)\right) + \left(\hat{g}(x, \theta_g) - \hat{g}(x, \theta_g^*)\right) u - \nu\right] \tag{3.45}$$

Based on (3.45), if we minimize the quadratic cost functional

$$\mathcal{J}_{ss}(\theta_f, \theta_g) = \frac{\alpha^2}{2}|e_{ss}|^2$$

subject to $|\theta_f| \leq M_f$, $|\theta_g| \leq M_g$, by using the gradient projection method, we obtain weight adjustment laws of the form (3.43), (3.44), with the exception that e_x is replaced by its quasi-steady-state response e_{ss}. This indicates the close relationship between dynamic backpropagation-type algorithms [18]-[20] and the adaptive laws described in this paper. $\diamond$

The next theorem establishes stability of the proposed identification scheme.

Theorem 4 *Consider the error filtering identification model (3.34). The adaptive laws (3.43), (3.44) guarantee that*

- e_x, $\hat{x}$, ϕ_f, $\phi_g \in L_\infty$.

- *there exist constants k_1, k_2 such that*

$$\int_0^t |e_x(\tau)|^2 d\tau \;\; \leq \;\; k_1 \; + \; k_2 \int_0^t (\delta_f + \delta_g + |\nu(\tau)|)^2 \, d\tau$$

Proof: Consider the Lyapunov function candidate

$$V(e_x, \phi_f, \phi_g) = \frac{1}{2} e_x^T e_x + \frac{1}{2\gamma_f} \phi_f^T \phi_f + \frac{1}{2\gamma_g} \phi_g^T \phi_g \tag{3.46}$$

Using (3.42), (3.43), (3.44), the time derivative of V in (3.46) can be expressed as

$$\dot{V} = -\alpha |e_x|^2 + e_x^T f_0 + e_x^T g_0 u - e_x^T \nu + I_1^* \frac{\theta_f^T \xi^T e_x}{|\theta_f|^2} \theta_f^T \phi_f + I_2^* \frac{\theta_g^T \zeta^T e_x u}{|\theta_g|^2} \theta_g^T \phi_g \tag{3.47}$$

where the last two terms in (3.47) are due to the projection modification. Using the same procedure as in Lemma 1, it can be shown that these terms are non-positive. Therefore (3.47) becomes

$$\begin{aligned} \dot{V} \;\; & \leq \;\; -\alpha |e_x|^2 + |e_x||f_0| + u_0 |e_x||g_0| + \nu_0 |e_x| \\ & \leq \;\; -\alpha |e_x|^2 + \delta_f |e_x||\phi_f| + \delta_g u_0 |e_x||\phi_g| + \nu_0 |e_x| \end{aligned} \tag{3.48}$$

where u_0, ν_0 are the bounds for $u(t)$ and $\nu(t)$ respectively, and δ_f, δ_g are as defined in Equations (3.37), (3.41) respectively. Since $\theta_f \in \mathcal{B}(M_f)$, $\theta_g \in \mathcal{B}(M_g)$, it is clear that $|\phi_f| \leq 2M_f$, $|\phi_g| \leq 2M_g$; hence, (3.48) can be written in the form

$$\dot{V} \leq -\alpha |e_x|^2 + \beta |e_x| \tag{3.49}$$

where $\beta := 2\delta_f M_f + 2u_0 \delta_g M_g + \nu_0$. Therefore, for $e_x > \beta/\alpha$, we have $\dot{V} < 0$ which implies that $e_x \in L_\infty$ and consequently $\hat{x} \in L_\infty$. The proof of the second part follows along the same lines as its counterpart in Theorem 2. □

Remark 3.7: To retain the generality of the result, in the above analysis we have not assumed any special structure for $\hat{f}(x, \theta_f)$ and $\hat{g}(x, \theta_g)$ except that these functions are sufficiently smooth. Given a specific number of layers and type of sigmoidal nonlinearity, estimates of δ_f, δ_g can be computed. ◇

Remark 3.8: As a consequence of the weights appearing non-affinely, the regressor filtering and optimization techniques descibed in Section 3.1.2 for RBF networks, cannot be applied, at least directly, in the case that the identification model is based on multilayer networks. ◇

4 Control

The control of dynamical systems deals with the problem of designing a controller that causes the system to display some desired behavior. This desired behavior is specified in terms of the control objectives which may include: maintaining the output of the system around some desired constant value (*regulation*); ensuring that the outputs follow some desired trajectory (*tracking*); forcing the outputs of the system to follow the outputs of a prespecified reference model (*model reference control*), or assuring that the overall system minimizes some performance criterion (*optimal control*). The problem of controlling a general nonlinear system is a formidable task even when the dynamics of the system are completely known. This difficulty is reflected in the fact that it was not until recently that any systematic procedure for the synthesis and analysis of a reasonably large class of nonlinear control systems was found. This class of nonlinear systems, which is referred to as *feedback linearizable*, or simply, *linearizable* systems, has attracted considerable attention among system and control theorists [5, 6].

In many areas of science and engineering we encounter systems for which the determination of an accurate model, based on the physical properties of the system, is not possible. In this section we are concerned with the problem of controlling such systems. In particular, we consider nonlinear dynamical system with unknown, or partially unknown nonlinearities. The objective is to approximate the unknown nonlinear functions by neural networks and develop adaptive laws for on-line adjustment of the weights of these networks, such that the stability of the overall system is guaranteed, and furthermore, the control objective is asymptotically achieved. This problem is closely related to the areas of adaptive linear control [2], and adaptive nonlinear control [56], where the uncertainty in the system is due to some unknown parameters. Allowing unknown nonlinearities in the system to be controlled, extends the class of uncertain systems that adaptive control techniques can be applied to, but at the same time, increases considerably the complexity in the analysis of the overall control system behavior. Due to the intricacies of the problem, it is very crucial to first understand what are the possible circumstances that may lead to instability and how these can be prevented.

4.1 Instability Mechanisms

In this subsection we investigate various types of instability mechanisms that may arise in control of nonlinear systems using neural network approximators and propose modifications to the standard adaptive and control laws for dealing with these problems. The instability discussion is motivated by the first order system

$$\dot{x} = f(x) + g(x)u \tag{4.1}$$

where $u \in \mathcal{R}$ is the control input, $x \in \mathcal{R}$ is the state, which is available for measurement, and f, g are unknown smooth functions. It will be assumed (without loss of generality) that $x = 0$ is an equilibrium point of the undriven system, i.e $f(0) = 0$. The control objective is to force the plant output x to follow the output x_m of a *reference model*

$$\dot{x}_m = -a_m x_m + b_m r \tag{4.2}$$

where $a_m > 0$, b_m are constants and $r(t)$ is a bounded reference input. Although simple, this first order system captures many of the difficulties that are encountered in the problem of controlling nonlinear systems using neural networks, or more generally, using approximations for the unknown nonlinearities.

If f and g were known and $g(x) \neq 0$ for all x in some region $\mathcal{N}_c(0)$ around the equilibrium $x = 0$, then it is easy to verify that the control law

$$u = \frac{1}{g(x)} \left[-a_m x + b_m r - f(x) \right] \tag{4.3}$$

would achieve the control objective asymptotically. More precisely, in this case the tracking error $e = x_m - x$ satisfies $e(t) \equiv 0$ if $x(0) = x_m(0)$, or satisfies $\lim_{t\to\infty} e(t) = 0$ if $x(0) \neq x_m(0)$. The requirement $g(x) \neq 0, \quad \forall x \in \mathcal{N}_c(0)$ is a *controllability* condition [57]; i.e, for $g(x(t)) = 0$ the system is no longer controllable.

Since f, g are unknown, we will approximate these functions by neural network models with outputs $\hat{f}(x, \theta_f)$, $\hat{g}(x, \theta_g)$ respectively, where θ_f and θ_g are weight vectors which are to be adjusted according to some adaptive rule. A straightforward choice for the control law is the *certainty equivalence controller* [4], obtained by replacing the unknown functions $f(x)$ and $g(x)$ in (4.3) by their estimates $\hat{f}(x, \theta_f)$ and $\hat{g}(x, \theta_g)$ respectively. This gives the control law

$$u \; = \; u_c \; = \; \frac{1}{\hat{g}(x, \theta_g)} \left[-a_m x + b_m r - \hat{f}(x, \theta_f) \right] \tag{4.4}$$

where, for the time being, it is assumed that the weights θ_g are such that $\hat{g}(x, \theta_g)$ is bounded away from zero. Using the control law (4.4) in (4.1), it can be shown after some manipulation that $e = x_m - x$ satisfies the following differential equation:

$$\dot{e} = -a_m e + [\hat{f}(x, \theta_f) - f(x)] + [\hat{g}(x, \theta_g) - g(x)] u_c \qquad e(0) = x_m(0) - x(0) \tag{4.5}$$

Following the same procedure as in Section 3, we will let θ_f^*, θ_g^* be the optimal weights, in the sense that they minimize $|f(x) - \hat{f}(x, \theta_f^*)|$ and $|g(x) - \hat{g}(x, \theta_g^*)|$ respectively, for $x \in \mathcal{X}$ and subject to the constraints $|\theta_f^*| \leq M_f$, $|\theta_g^*| \leq M_g$, where M_f, M_g are design constants and $\mathcal{X} \subset \mathcal{R}$ is some compact set. More conditions on the properties of $\mathcal{X}$ are discussed later.

Therefore, (4.5) is rewritten in terms of the (unknown) optimal weights as

$$\dot{e} = -a_m e + \left[\hat{f}(x, \theta_f) - \hat{f}(x, \theta_f^*) \right] + \left[\hat{g}(x, \theta_g) - \hat{g}(x, \theta_g^*) \right] u_c + \nu(t) \tag{4.6}$$

where $\nu(t)$ represents the modelling error, which is given by

$$\nu(t) = \left[\hat{f}(x(t), \theta_f^*) - f(x(t)) \right] + \left[\hat{g}(x(t), \theta_g^*) - g(x(t)) \right] u_c(t)$$

It is worth noting that (4.6) is of the same form as the error equation of the identification model developed in Section 3, based on the error filtering scheme. In identification, the constant $-\alpha$ was the pole location of the stable filter, while in control, the corresponding constant $-a_m$ is the pole location of the reference model. A major difference between the two schemes is the role of the input signal. In identification, u is an external bounded signal that is chosen by the designer and fed to both the identification model and the real system, which is stable. On the other hand, in the control procedure the signal u_c is the certainty equivalence control law given by (4.4), which is composed of signals from the output of the system (*feedback*), as well as the output of the neural network estimation model (*adaptation*).

Based on the error equation (4.6), one may consider the adaptive laws developed in Section 3 for continuous adjustment of the weights. However, in order to guarantee stability of the overall scheme, we have to impose additional assumptions or consider modifications to the adaptive laws. In identification the problem was simplified in two respects. First, the input signal was chosen by the designer; hence it could be readily assumed that only bounded input signals were used. In control, the boundedness of u_c cannot be guaranteed apriori. Second, in the identification problem we considered only stable systems, thus ensuring that the output of the system was bounded. In control problems, due to the feedback control law, the boundedness of the output cannot be guaranteed even if the system is originally stable.

Based on these observations, we are faced with the following three possible types of instability problems.

(a) Parameter Drift : This kind of problem was also encountered (and remedied) in the identification schemes described in Section 3. The characteristic of parameter drift is the tendency of weights to drift to infinity with time. In the case of neural network models, the parameter drift is caused by the modelling error, which enters the identification or control scheme by the inadequacy of a neural network to match exactly the nonlinear function that it tries to approximate, even if an optimal set of weights were to be chosen. This mismatch between the output of an "optimal" static network model and a given nonlinear function could be due to many factors but, a principal cause is insufficient number of weights. Of course, for special classes of functions, it is possible that no modelling error is present. For example, the scalar function

$$f(x) = 3e^{(x-2)^2}$$

can be modelled *exactly* for all $x \in \mathcal{R}$ by a Gaussian RBF network with a single unit, simply by choosing the center $c_1 = 2$, the width $\sigma_1 = 1$ and the weight $w_1 = 3$. However, for a general function this will not be the case and therefore the modelling error is something that has to be dealt with.

In Section 3 we described the projection algorithm modification, one method of preventing parameter drift. The basic idea behind the projection algorithm is as follows: first choose the initial weight values to belong to some (large) compact set that is chosen apriori; then, the weight adjustment proceeds according to the standard adaptive law (obtained by the Lyapunov or optimization method), unless the weights are directed towards leaving the compact set. In this case, the weight values are projected back onto the compact set by a projection algorithm.

Although there are other techniques for preventing parameter drift, the projection algorithm has the natural and intuitive interpretation of confining the weights to some compact set that is chosen by the designer. Another major advantage of using the projection algorithm for dealing with the parameter drift problem is the fact that no apriori information regarding the bound on the modelling error is required. Furthermore, from a practical viewpoint, restricting the weight values from becoming too large, can prevent many problems that may otherwise occur due to numerical errors.

(b) Controllability Problem : In defining the certainty equivalence controller u_c, we have made the crucial assumption that the weights θ_g are such that $\hat{g}(x, \theta_g)$ is bounded away from zero. It is clear that a standard adaptive law for generating $\theta_g(t)$ (obtained by the Lyapunov or optimization method) does not guarantee apriori that this will be true. Therefore some kind of modification in the update law for θ_g is required in order to ensure that controllability of the estimation model is retained. In general, this is far from being a trivial task; adaptive control theorists are aware, from the linear case, that this is one of the major problems with indirect-type of control schemes [58].

In this paper we take the approach of attempting to find a convex subset $\mathcal{K}_g$ of the θ_g parameter space such that $\theta_g^* \in \mathcal{K}_g$ and for every $\theta_g \in \mathcal{K}_g$ we have $|\hat{g}(x,\theta_g)| \geq \delta > 0$, where δ is a small constant. If such set can be found, then controlability based on the estimated parameters $\theta_g(t)$ at each time t is ensured by incorporating a projection algorithm [50, 54] (of the type described in Section 3) in the adaptive law, to guarantee that $\theta_g(t) \in \mathcal{K}_g$ for all $t \geq 0$. Following this procedure, the usual properties of the adaptive law (e.g, stability) are not altered. This approach, although simple, relies on the strong requirement that the set $\mathcal{K}_g$ is known. We next describe how to obtain such a set $\mathcal{K}_g$ for the two cases that $\hat{g}(x,\theta_g)$ is the output of: 1) a Gaussian RBF network, and 2) a sigmoidal multilayer network. Without loss of generality, we will assume that $g(x)$ is positive[5] for all $x \in \mathcal{N}_c(0)$.

In the case of Gaussian RBF network models, the output of the network has the form $\hat{g}(x,\theta_g) = \theta_g^T \xi(x)$, where $\xi \in \mathcal{R}^{n_g}$ is a vector of Gaussian functions; therefore each element of the vector $\xi(x)$ is positive for all x. As a consequence, if each element of θ_g is positive then $\theta_g^T \xi(x)$ will also be positive. Based on this observation, $\mathcal{K}_g$ is chosen as

$$\mathcal{K}_g := \{\theta \in \mathcal{R}^{n_g} \;:\; \theta_i > \delta, \;\; i = 1, \cdots n_g\} \tag{4.7}$$

where δ is a small constant. Even though it is not necessarily true (most of the times it will be true) that $\theta_g^* \in \mathcal{K}_g$, based on the same technique as in Section 3, we can define the optimal weights θ_g^* under the additional constraint that $\theta_g^* \in \mathcal{K}_g$. Since $g(x)$ is positive, any modelling error introduced by restricting θ_g^* to be in $\mathcal{K}_g$, will be small.

In the case of sigmoidal multilayer neural network models, we can obtain a "controllability set" $\mathcal{K}_g$ in the same way by expressing $\hat{g}(x,\theta_g)$ as

$$\hat{g}(x,\theta_g) = \bar{\theta}_g^T z(x,\tilde{\theta}_g)$$

where z is the output of the *last* hidden layer, the vector $\bar{\theta}_g$ represents the weights of the last layer and $\tilde{\theta}_g$ represents the weights of the rest of the layers. In this formulation, $\theta_g \in \mathcal{R}^{n_1+n_2}$ is broken up into $\bar{\theta}_g \in \mathcal{R}^{n_1}$ and $\tilde{\theta}_g \in \mathcal{R}^{n_2}$. If the sigmoidal nonlinearity $\sigma(\cdot)$ that is used is such that, $\sigma(p)$ is strictly positive for all p, then each element of the vector $z(x,\tilde{\theta}_g)$ will be positive for all $(x,\tilde{\theta}_g)$. A particular type of sigmoid whose output is always positive is the logistic function $\sigma(p) := (1+e^{-p})^{-1}$. Following the same procedure as in the case of RBF networks, the set $\mathcal{K}_g$ can therefore be defined as

$$\mathcal{K}_g := \{\theta \in \mathcal{R}^{n_1+n_2} \;:\; \bar{\theta}_i > \delta, \;\; i = 1, \cdots n_1\}$$

[5]In general, we need to know the sign of $g(x)$. This is a requirement that is needed even in the adaptive control of linear systems [2].

where $\bar{\theta}$ is to interpreted as representing the weights of the last layer and δ is again a small constant.
(c) Transient Behavior Problem : A common characteristic of adaptive systems is that large initial parameter estimation errors result in bad transient behavior. Intuitively, this is to be expected since large initial parameter estimation errors, in general imply that (initially) there is a large discrepancy between the functions $f(x)$, $g(x)$ and their approximations $\hat{f}(x, \theta_f)$ and $\hat{g}(x, \theta_g)$ respectively, which in turn may cause large tracking errors in the early stages of adaptation.

In the context of using neural network architectures to approximate the unknown functions $f(x)$ and $g(x)$, bad transient behavior may cause additional problems. In particular, adaptive laws for adjustment of the weights are developed under the assumption that $x(t)$ remains in some compact set $\mathcal{X}$ for all $t \geq 0$, where $\mathcal{X}$ represents the region that the approximation of the functions $f(x)$ and $g(x)$ is considered. Hence, if $x(t)$ leaves the region $\mathcal{X}$ due to bad transient behavior then this would, in general, increase the modelling error, which could force $x(t)$ even further away from $\mathcal{X}$, eventually causing instability. It is noted that in the identification schemes developed in Section 3, there was not such problem because the system was assumed to be stable and adaptation was performed off-line.

One way of dealing with this problem is to first use some identification procedure so that during the control phase, the estimates $\hat{f}(x, \theta_f)$, $\hat{g}(x, \theta_g)$ represent "good" approximations of the functions $f(x)$ and $g(x)$ respectively. However, this would only work for systems that are already stable, or systems for which we have enough apriori information to stabilize it before we employ approximation techniques for better performance.

Another way of dealing with the transient behavior problem was suggested by Sanner et.al. [28]. According to [28], an auxiliary component is added in the control law, whose purpose is to introduce a *sliding control* term for confining $x(t)$ to the region $\mathcal{X}$. The sliding control methodology [59], which is sometimes used as an alternative to adaptive control in systems with parametric uncertainty, represents a rather simple approach to robust control with applications in robot manipulators, power systems, and others [60]. The principal idea behind using sliding control in the context of this paper, is very similar to that of the projection algorithm, which was introduced earlier; in both cases the objective is to restrict a signal or a parameter value from entering an undesired region. Since the designer can directly influence the weight values, the projection algorithm can be employed for confining the weights to a certain region. On the other hand, if the objective is to restrict the output x in a certain region then this has to be done indirectly through the control signal u, which is exactly what the sliding control term attempts to do.

Therefore from this point of view, projection provides a direct way of confining weight values, while sliding control provides an indirect method for confining the output signal. Due to its indirect way of affecting the output signal, the sliding technique requires considerable apriori information concerning the unknown plant, in the form of bounds on $f(x)$ and $g(x)$. Another drawback of the sliding control design is the extremely high control effort that it requires. Typically, high control activity is not only expensive but it may also excite any unmodelled dynamics of the system [60]. In fact, sliding control in this context, is analogous

Figure 8: A block diagram representation of the control scheme given by (4.8) with neural network models as estimators of the unknown nonlinearities.

to high-gain feedback [61], which is a classical control tool for reducing the effects of unknown disturbances. A detail design procedure of the sliding control term is given in the next subsection.

4.2 Stable Control of First Order Systems

Using the modifications discussed in Section 4.1 for dealing with the paramater drift, controllability and transient behavior problems, we now consider the design of stable control laws for first order systems of the form described by (4.1). These results are extended to a class of higher order systems in the next subsection. For simplicity, we examine the case where the unknown nonlinearities $f(x)$ and $g(x)$ are modelled by RBF networks. As in Section 3, the same approach can be applied to multilayer networks with sigmoidal type of nonlinearities; however, the results obtained will be weaker, in the sense that, even if there is no modelling error, it cannot be guaranteed that the tracking error converges to zero.

Following the discussion of Section 4.1, the certainty equivalence control u_c, given by (4.4), is appended by the sliding control term u_s. Therefore we have

$$\begin{aligned} u &= u_c + u_s \\ &= \frac{1}{\theta_g^T \zeta(x)} \left[-a_m x + b_m r - \theta_f^T \xi(x) \right] + u_s \end{aligned} \tag{4.8}$$

where $\theta_f \in \mathcal{R}^{n_f}$, $\theta_g \in \mathcal{R}^{n_g}$ represent the adjustable weights and ξ, ζ are again the outputs of Gaussian basis functions in the approximation of $f(x)$ and $g(x)$ respectively. This controller structure leads to the scheme shown in Figure 8.

Design of u_s : Since u_s is to be used for confining $x(t)$ in $\mathcal{X}$ for all $t \geq 0$, it is important that $\mathcal{X}$ be chosen appropriately. From a theoretical viewpoint, the choice of $\mathcal{X}$ is required only for the design of u_s. However, from a practical perspective, $\mathcal{X}$ also represents the region in which the approximation is to be performed. In the case of RBF networks, this is the region where the centers of the basis functions are distributed. For ease of discussion we take $\mathcal{X}$ to be a ball of radius M_x, i.e, $\mathcal{X} = \mathcal{B}(M_x)$. From (4.2), the output x_m of the reference model satisfies

$$\begin{aligned} |x_m(t)| &\leq e^{-a_m t}|x_m(0)| + \int_0^t e^{-a_m(t-\tau)} r(\tau) d\tau \\ &\leq |x_m(0)| + \frac{r_0}{a_m} \quad := \quad M_r \end{aligned}$$

where r_0 is an upper bound for $|r(t)|$. Since the control objective is to track x_m, it is clear that M_x should be greater than M_r. We will let $M_x = 3M_r$. At this stage it has to be assumed that $\mathcal{B}(M_x) \subset \mathcal{N}_c(0)$, where, it is recalled, $\mathcal{N}_c(0)$ is the controllability region of the real system. It is also assumed that $|x(0)| \leq \frac{1}{3}M_x$. This last assumption is made without any loss of generality since M_x can be chosen large enough such that $|x(0)| \leq \frac{1}{3}M_x$ is satisfied.

Using the composite control law (4.8), it can be shown after some straightforward rearrangement of terms that the tracking error $e = x_m - x$ satisfies the differential equation

$$\dot{e} = -a_m e + \left[\theta_f^T \xi - f(x)\right] + \left[\theta_g^T \zeta - g(x)\right] u_c - g(x) u_s \tag{4.9}$$

Based on (4.9) consider the Lyapunov function $V_e(t) = \frac{1}{2}e^2(t)$. Then

$$\begin{aligned} \dot{V}_e &= -a_m e^2 + e\left[\theta_f^T \xi - f(x)\right] + e\left[\theta_g^T \zeta - g(x)\right] u_c - e g(x) u_s \\ &\leq -a_m e^2 + \left[|\theta_f^T \xi| + |f| + |\theta_g^T \zeta u_c| + |g u_c|\right] |e| - g e u_s \end{aligned} \tag{4.10}$$

Suppose u_s is of the form $u_s = q(t)\,\mathrm{sgn}\{e\}$ where $\mathrm{sgn}\{e\} = 1$ if $e \geq 0$ and $\mathrm{sgn}\{e\} = 0$ if $e < 0$. Equation (4.10) can now be expressed as

$$\begin{aligned} \dot{V}_e &\leq -a_m e^2 + \left[|\theta_f^T \xi| + |f| + |\theta_g^T \zeta u_c| + |g u_c|\right] |e| - g q e\, \mathrm{sgn}\{e\} \\ &\leq -a_m e^2 + \left[|\theta_f^T \xi| + |f| + |\theta_g^T \zeta u_c| + |g u_c| - g q\right] |e| \end{aligned} \tag{4.11}$$

From (4.11), if $q(t)$ is chosen such that the inequality

$$q(t) \geq \frac{1}{g(x(t))}\left[\, \left|\theta_f^T(t)\xi(x(t))\right| + |f(x(t))| + \left|\theta_g^T(t)\zeta(x(t))u_c(t)\right| + |g(x(t))u_c(t)| \,\right] \tag{4.12}$$

is satisfied for all $t \geq 0$, then $\dot{V}_e \leq -a_m e^2$, which implies that the tracking error $e(t)$ is decreasing. In order to choose a function $q(t)$ that satisfies the inequality (4.12), we need to assume that we know upper bounds on $f(x)$ and $g(x)$, and a lower bound on $g(x)$; these bounds should be valid for all $x \in \mathcal{B}(M_x)$. Let $f^u(x)$, $g^u(x)$ be known state dependent upper bounds on $f(x)$ and $g(x)$ respectively and also let $g_l(x)$ be a known lower bound on $g(x)$; i.e, $0 < g_l(x) \leq g(x)$ for all $x \in \mathcal{B}(M_x)$.

Using state dependent bounds is less restrictive than requiring a fixed bound for all $x \in \mathcal{B}(M_x)$. Nevertheless, having knowledge of these bounds is a very restrictive assumption that limits the applicability of these techniques. It is recalled that these assumptions are required in the design of the sliding controller, whose purpose is to confine $x(t)$ within the region $\mathcal{X}$. Therefore, these requirements are more of a safeguard rather than an integral part of the design. Based on the above bounds, $q(t)$ is chosen as

$$q(t) := \frac{1}{g_l(x(t))}\left[\left|\theta_f^T(t)\xi(x(t))\right| + |f^u(x(t))| + \left|\theta_g^T(t)\zeta(x(t))u_c(t)\right| + |g^u(x(t))u_c(t)|\right] \quad (4.13)$$

The action of the sliding control is required only when there is a danger that x might leave the region $\mathcal{B}(M_x)$. Let $I_{u_s}^*$ be the indicator function defined as $I_{u_s}^* = 1$ if $|e| \geq \frac{2}{3}M_x$ and $I_{u_s}^* = 0$ if $|e| < \frac{2}{3}M_x$. The sliding control term is chosen as

$$u_s = I_{u_s}^* q(t)\,\mathrm{sgn}\{e\} \quad (4.14)$$

To verify that this choice for u_s guarantees that $|x(t)| \leq M_x$ for all $t \geq 0$, we first note that since by assumption $|x(0)| \leq \frac{1}{3}M_x$, we have

$$\begin{aligned} |e(0)| &\leq |x(0)| + |x_m(0)| \\ &\leq \frac{1}{3}M_x + M_r = \frac{2}{3}M_x \end{aligned}$$

Now, if $|e(t)| = \frac{2}{3}M_x$ then, from (4.11) and (4.13), we have that $|e(t)|$ is decreasing. This implies that $|e(t)| \leq \frac{2}{3}M_x$ for all $t \geq 0$. Therefore using the inequality

$$\begin{aligned} |x(t)| &\leq |e(t)| + |x_m(t)| \\ &\leq \frac{2}{3}M_x + M_r = M_x \end{aligned}$$

we conclude that $x(t) \in \mathcal{B}(M_x)$ for all $t \geq 0$.

One potential problem is the fact that u_s, as given by (4.14), is discontinuous with respect to x. Discontinuous control laws not only create problems of existence and uniqueness of solutions [51], but are also known to display chattering phenomena [59] and excite high frequency unmodelled dynamics [60]. One way of avoiding these problems is to approximate the discontinuous terms in (4.14) by sigmoidal functions . As pointed out in [28], if these sigmoidal functions are made "steep enough" then they represent good approximations of the discontinuous functions, thus resulting in a similar behavior.

Design of the adaptive laws : We now turn our attention towards designing weight adjustment laws for θ_f and θ_g. Let the weight estimation errors be defined as $\phi_1(t) := \theta_f(t) - \theta_f^*$ and $\phi_2(t) := \theta_g(t) - \theta_g^*$. The optimal weight set θ_f^* is equal to the value of θ that minimizes $|f(x) - \theta^T\xi(x)|$ for $x \in \mathcal{X}$ and subject to the constraint $\theta \in \mathcal{B}(M_f)$, while θ_g^* is equal to the value of θ that minimizes $|g(x) - \theta^T\zeta(x)|$ for $x \in \mathcal{X}$ and subject to the constraint

$\theta \in \mathcal{B}(M_f) \cap \mathcal{K}_g$. The set $\mathcal{K}_g$, which is given by (4.7), characterizes a subset of the θ_g weight space for which controllability of the estimation model is ensured.

Based on the above definitions for the optimal weights θ_f^*, θ_g^*, we start by adding and subtracting $\theta_f^{*T}\xi + \theta_g^{*T}\zeta u_c$ in the error equation (4.9). This yields

$$\dot{e} = -a_m e + \phi_1^T \xi + \phi_2^T \zeta u_c - \nu - g u_s \tag{4.15}$$

where the modelling error ν is given by

$$\nu(t) = \left[f(x(t)) - \theta_f^{*T}\xi(x(t))\right] + \left[g(x(t)) - \theta_g^{*T}\zeta(x(t))\right] u_c(t) \tag{4.16}$$

In developing adaptive laws for θ_f and θ_g, it is convenient to define the indicator functions I_f^* and I_g^* as follows: if θ_f is on the boundary of $\mathcal{B}(M_f)$ and directed outwards then $I_f^* = 1$; otherwise $I_f^* = 0$. Correspondingly, $I_g^* = 1$ if θ_g is on the boundary of the set $\mathcal{B}(M_g) \cap \mathcal{K}_g$ and directed towards leaving the set, and $I_g^* = 0$ otherwise. Utilizing these indicator functions, the weights are continuously adjusted according to the following adaptive laws:

$$\dot{\theta}_f = \begin{cases} -\gamma_1 e \xi & \text{if} \quad I_f^* = 0 \\ \mathcal{P}\left\{-\gamma_1 e \xi\right\} & \text{if} \quad I_f^* = 1 \end{cases} \tag{4.17}$$

$$\dot{\theta}_g = \begin{cases} -\gamma_2 e \zeta u_c & \text{if} \quad I_g^* = 0 \\ \mathcal{P}\left\{-\gamma_2 e \zeta u_c\right\} & \text{if} \quad I_g^* = 1 \end{cases} \tag{4.18}$$

The adaptive gains γ_1, γ_2 are positive constants and $\mathcal{P}\{-\gamma_1 e\xi\}$ denotes the projection of $-\gamma_1 e\xi$ onto the supporting hyperplane to the convex set $\mathcal{B}(M_f)$ at θ_f (and correspondingly for θ_g). The initial weights $\theta_f(0)$, $\theta_g(0)$ are chosen to belong to the sets $\mathcal{B}(M_f)$ and $\mathcal{B}(M_g) \cap \mathcal{K}_g$ respectively.

Theorem 5 *Consider the first order system (4.1) with the composite control law (4.8), (4.14) and the adaptive laws (4.17), (4.18). The overall control scheme guarantees the following properties:*

- $e, x, u, \phi_1, \phi_2 \in L_\infty$.
- *there exist constants k_1, k_2 such that*

$$\int_0^t |e(\tau)|^2 d\tau \;\leq\; k_1 + k_2 \int_0^t |\nu(\tau)|^2 d\tau$$

Proof: Consider the Lyapunov function candidate

$$V(e, \phi_1, \phi_2) = \frac{1}{2}e^2 + \frac{1}{2\gamma_1}\phi_1^T\phi_1 + \frac{1}{2\gamma_2}\phi_2^T\phi_2 \tag{4.19}$$

It can be readily verified that the time derivative of V satisfies

$$\dot{V} = -a_m e^2 - e\nu - I_{u_s}^* \, g(x)q(t)|e| + I_f^* \, \pi_f(e, \xi, \theta_f) + I_g^* \, \pi_g(e, \zeta, u_c, \theta_g) \tag{4.20}$$

where $I_f^* \pi_f$ and $I_g^* \pi_g$ are additional terms that enter (4.20) due to the projection algorithm modification in the adaptive laws. Using the same procedure as in Lemma 1, it can be shown that these terms are non-positive. Furthemore, since $g(x)$ and $q(t)$ are positive, the term $-I_{u_s}^* g(x)q(t)|e|$ is also non-positive. Therefore (4.20) becomes

$$\dot{V} \leq -a_m e^2 - e\nu \tag{4.21}$$

The projection modification in the adaptive laws ensures that $|\theta_f| \leq M_f$, $|\theta_g| \leq M_g$. Therefore $\phi_1, \phi_2 \in L_\infty$. Also, the sliding mode control term u_s in (4.8) guarantees that $|x(t)| \leq M_x$ for all $t \geq 0$. This together with the fact that $\theta_g(t) \in \mathcal{K}_g$ for $t \geq 0$, implies that $u_c \in L_\infty$. Therefore there exists a finite constant ν_0 such that $\sup_{t\geq 0} |\nu(t)| \leq \nu_0$. Hence from (4.21) we obtain that $e \in L_\infty$. Thus all the signals in the adaptive control scheme remain bounded. The second part of the theorem follows directly from (4.21) along the same lines as in Theorem 2. □

Remark 4.1: Whether or not the tracking error converges to zero, depends on how large is the modelling error. In the ideal case that $\nu(t) = 0$ then $e \in L_2$ and hence using the fact that $\dot{e} \in L_\infty$, we obtain (using Barbalat's Lemma) that $\lim_{t\to\infty} e(t) = 0$. If $\nu(t)$ is not identically zero but is square integrable (i.e, $\nu \in L_2$), then from the second part of Theorem 5 we have that $e \in L_2$ and hence it is concluded again that $e(t)$ converges to zero. ◇

4.3 Extension to nth Order Systems

The results obtained in Section 4.2 for the first order system (4.1) can be extended in a straightforward manner to nth order systems of the form

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= x_3 \\ &\vdots \\ \dot{x}_n &= f(x_1, x_2, \cdots, x_n) + g(x_1, x_2, \cdots, x_n)u \\ y &= x_1 \end{aligned} \tag{4.22}$$

or equivalently of the form

$$x^{(n)} - f(x, \dot{x}, \ddot{x}, \cdots x^{(n-1)}) = g(x, \dot{x}, \ddot{x}, \cdots x^{(n-1)})u\ , \qquad y = x \tag{4.23}$$

We will refer to systems of the form (4.22), or equivalently (4.23), as being in *canonical* form. In the spirit of the feedback linearization literature (see [5, 6]), these systems are in normal form and have no zero dynamics.

We consider the case that the control objective is to force y to follow the output y_m of the reference model

$$\begin{aligned} \dot{x}_m &= Ax_m + br \\ y_m &= c^T x_m \end{aligned} \tag{4.24}$$

where A is a Hurwitz $n \times n$ matrix. In order to have a well-posed problem, it is assumed that the relative degree of the reference model is equal to n.[6] This implies [57]

$$c^T b = c^T A b = \cdots = c^T A^{n-2} b = 0 \tag{4.25}$$

If we let $X = [x_1, x_2, \cdots x_n]^T$, then using (4.22), (4.25), it can be readily verified that the tracking error $e = y_m - y$ satisfies

$$\begin{aligned} e^{(n)} &= y_m^{(n)} - y^{(n)} \\ &= c^T A^n x_m + c^T A^{n-1} b r - f(X) - g(X) u \end{aligned} \tag{4.26}$$

Let $k = [k_n,\ k_{n-1},\ \cdots\ k_1]^T$, where the constants $\{k_i : i = 1, \cdots n\}$ are such that all the roots of the polynomial $h(s) = s^n + k_1 s^{n-1} + \cdots + k_n$ are in the open left half plane. Also let $\varepsilon := \left[e,\ \dot{e},\ \ddot{e}, \cdots\ e^{(n-1)}\right]$. Using (4.25), each element of the vector ε can be measured according to

$$e^{(i)} = c^T A^i x_m - x_{i+1} \qquad\qquad i = 0, 1, \cdots n-1$$

First suppose that the functions f and g are known and $g(X) \neq 0$ for $X \in \mathcal{N}_c$, where $\mathcal{N}_c \subset \mathcal{R}^n$ is the controllability region. The control law

$$u = \frac{1}{g(X)} \left[-f(X) + c^T A^n x_m + c^T A^{n-1} b r + k^T \varepsilon\right] \tag{4.27}$$

applied to the error system (4.26) results in

$$e^{(n)} = k^T \varepsilon$$

or equivalently

$$h(s)[\varepsilon] = 0$$

This implies $\lim_{t \to \infty} e(t) = 0$, which achieves the control objective.

Since f and g are unknown, we replace these functions in the control law (4.27) by neural network models with output $\hat{f}(X, \theta_f)$ and $\hat{g}(X, \theta_g)$ respectively. Therefore the certainty equivalence controller $u = u_c$ is given by

$$u_c = \frac{1}{\hat{g}(X, \theta_g)} \left[-\hat{f}(X, \theta_f) + c^T A^n x_m + c^T A^{n-1} b r + k^T \varepsilon\right] \tag{4.28}$$

By applying the control law (4.28) in (4.26), we obtain after some straightforward manipulation of terms the error equation

$$e^{(n)} = -k^T \varepsilon + \left[\hat{f}(X, \theta_f) - f(X)\right] + \left[\hat{g}(X, \theta_g) - g(X)\right] u_c \tag{4.29}$$

[6] The relative degree of the reference model should be equal to n because the relative degree of the plant (4.22) is equal to n. For details concerning the concepts of zero dynamics and relative degree of nonlinear systems, the reader is referred to [5].

or

$$h(s)[e] = \left[\hat{f}(X,\theta_f) - f(X)\right] + [\hat{g}(X,\theta_g) - g(X)]\,u_c$$

Equation (4.29) is now rewritten as

$$\dot{\varepsilon} = \Lambda_c \varepsilon + b_c\left[\left(\hat{f}(X,\theta_f) - f(X)\right) + (\hat{g}(X,\theta_g) - g(X))\,u_c\right] \tag{4.30}$$

where $\det(sI - \Lambda_c) = h(s)$, $\;b_c := [\,0\ \cdots\ 0\ 1\,]^T$ and $(\Lambda_c,\ b_c)$ is in controllable canonical form [57]. Once expressed in the form (4.30), then it is clear that we have the same error equation as (4.5), with the exception that in (4.5) the error e was a scalar, while in (4.30) the error ε is a vector. Therefore, based on (4.30), one can design control and adaptive laws using the same procedure as in Section 4.2. The stability analysis follows along the same lines as that of Section 4.2 and is therefore omitted.

Remark 4.2: The control procedure of this section was applied to a relatively simple class of nonlinear systems. Although similar techniques based on the certainty equivalence principle can be applied to a wider class of systems, the analysis of such control schemes becomes considerably more complicated. These difficulties are also present (to a lesser degree) in adaptive nonlinear control, where two types of control schemes have been considered [10] : *uncertainty-constrained schemes* impose restrictions (matching conditions) on the location of the unknown parameters, while *nonlinearity-constrained schemes* impose restrictions on the type of nonlinearities in the original system. We believe that in the future similar conditions will also be developed for systems with unknown nonlinearities in an attempt to enlarge the class of systems for which stable neural network techniques can be applied. $\diamond$

5 Concluding Remarks

The principal contribution of this paper is the synthesis and stability analysis of neural network based schemes for identifying and controlling dynamical systems with unknown nonlinearities. Two wide classes of neural network architectures have been considered: 1) multilayer neural networks with sigmoidal type of activation functions, and 2) RBF networks. Although these two classes of networks are evidently constructed differently, we have examined them in a common framework as approximators of the unknown nonlinearities of the system. Our results are intended to complement the numerous empirical results concerning learning and control that have appeared in the recent neural network literature. Furthermore, we have attempted to unify the learning techniques used by connectionists and adaptive control theorists. Bridging the gap between the two areas would certainly be beneficial to both fields.

APPENDIX A : Proof of Theorem 1

By Assumptions (S1), (S2), there exists a unique solution $x(t)$ to the differential equation (2.1) (with the initial condition $x(0) = x^0$) that satisfies $|x(t) - x^0| \leq k$ for all $t \in [0,T]$, where k is some positive constant. Let $\mathcal{K}$ be the compact set defined as

$$\mathcal{K} := \left\{(x,u) \in \mathcal{R}^{n+m}\ :\ |x - x^0| \leq k + \epsilon,\ u \in \mathcal{U}\right\}$$

By Assumption (N2), $\hat{g}$, $\hat{h}$ are continuous functions and therefore they satisfy a Lipschitz condition in the compact domain $\mathcal{K}$, i.e there exists constants l_g, l_h such that for all $(x_1, u), (x_2, u) \in \mathcal{K}$

$$\begin{aligned} |\hat{g}(x_1, u, \theta_g) - \hat{g}(x_2, u, \theta_g)| &\leq l_g|x_1 - x_2| && \text{(A.1)} \\ |\hat{h}(x_1, u, \theta_h) - \hat{h}(x_2, u, \theta_h)| &\leq l_h|x_1 - x_2| && \text{(A.2)} \end{aligned}$$

If we let $e_x := x - \hat{x}$ then from (2.1) and (2.3) we obtain

$$\dot{e}_x = Ae_x + g(x, u) - \hat{g}(\hat{x}, u, \theta_g^*) \qquad e_x(0) = 0 \qquad \text{(A.3)}$$

Based on Assumption (N1), the weight set θ_g^* in Equation (A.3) is chosen such that

$$\max_{(x,u)\in\mathcal{K}} \left| g(x, u) - \hat{g}(x, u, \theta_g^*) \right| \leq \epsilon_g \qquad \text{(A.4)}$$

where $\epsilon_g > 0$ is a constant to be chosen later. The solution to the differential equation (A.3) can be expressed as

$$\begin{aligned} e_x(t) &= \int_0^t e^{A(t-\tau)} \left[g(x(\tau), u(\tau)) - \hat{g}(\hat{x}(\tau), u(\tau), \theta_g^*) \right] d\tau \\ &= \int_0^t e^{A(t-\tau)} \left[g(x(\tau), u(\tau)) - \hat{g}(x(\tau), u(\tau), \theta_g^*) \right] d\tau \; + \\ &\qquad + \int_0^t e^{A(t-\tau)} \left[\hat{g}(x(\tau), u(\tau), \theta_g^*) - \hat{g}(\hat{x}(\tau), u(\tau), \theta_g^*) \right] d\tau \qquad \text{(A.5)} \end{aligned}$$

Since A is a Hurwitz matrix there exist positive constants c, α (that depend on A) such that $\|e^{At}\| \leq ce^{-\alpha t}$ for all $t \geq 0$. Based on the constants c, α, l_g, l_h, ϵ, let ϵ_g in (A.4) be chosen as

$$\epsilon_g := \frac{\epsilon\alpha}{2cl_h} e^{-cl_g/\alpha} > 0 \qquad \text{(A.6)}$$

First consider the case that $|\hat{x}(t) - x^0| < k + \epsilon$ for all $t \in [0, T]$, which implies that $(\hat{x}, u) \in \mathcal{K}$ in that interval. In this case, starting from (A.5), taking norms on both sides and using (A.1), (A.4), (A.6), the following inequalities hold for $t \in [0, T]$:

$$\begin{aligned} |e_x(t)| &\leq \int_0^t \|e^{A(t-\tau)}\| \cdot \left| g(x(\tau), u(\tau)) - \hat{g}(x(\tau), u(\tau), \theta_g^*) \right| \, d\tau \; + \\ &\qquad + \int_0^t \|e^{A(t-\tau)}\| \cdot \left| \hat{g}(x(\tau), u(\tau), \theta_g^*) - \hat{g}(\hat{x}(\tau), u(\tau), \theta_g^*) \right| d\tau \\ &\leq \int_0^t ce^{-\alpha(t-\tau)} \cdot \frac{\epsilon\alpha}{2cl_h} e^{-cl_g/\alpha} d\tau + \int_0^t ce^{-\alpha(t-\tau)} \cdot l_g |e_x(\tau)| d\tau \\ &\leq \frac{\epsilon}{2l_h} e^{-cl_g/\alpha} + cl_g \int_0^t e^{-\alpha(t-\tau)} |e_x(\tau)| d\tau \end{aligned}$$

Therefore by using the Bellman-Gronwall Lemma [45] we obtain

$$
\begin{aligned}
|e_x(t)| &\leq \frac{\epsilon}{2l_h} e^{-cl_g/\alpha} \cdot e^{cl_g \int_0^t e^{-\alpha(t-\tau)} d\tau} \\
|e_x(t)| &\leq \frac{\epsilon}{2l_h}
\end{aligned}
\tag{A.7}
$$

Note that from (A.2) it can be assumed without loss of generality that $l_h \geq 1$ and therefore (A.7) implies that $|\hat{x}(t) - x^0| \leq k + \epsilon/2$. Now suppose (for the sake of contradiction) that $(\hat{x}, u)$ does not belong to $\mathcal{K}$ for all $t \in [0, T]$. Then, by the continuity of $\hat{x}(t)$, there exists a T^*, where $0 < T^* < T$, such that $|\hat{x}(T^*) - x^0| = k + \epsilon$. Therefore if we carry out the same analysis for $t \in [0, T^*]$ we obtain that in this interval $|\hat{x}(t) - x^0| \leq k + \epsilon/2$, which is clearly a contradiction. Hence (A.7) holds for all $t \in [0, T]$.

Now consider the difference in outputs. By taking norms we obtain

$$
\begin{aligned}
|y(t) - \hat{y}(t)| &= |h(x, u) - \hat{h}(\hat{x}, u, \theta_h^*)| \\
&\leq |h(x, u) - \hat{h}(x, u, \theta_h^*)| + |\hat{h}(x, u, \theta_h^*) - \hat{h}(\hat{x}, u, \theta_h^*)|
\end{aligned}
$$

Again by Assumption (N1) we can choose θ_h^* such that

$$
\max_{(x,u) \in \mathcal{K}} \left| h(x, u) - \hat{h}(x, u, \theta_h^*) \right| \leq \frac{\epsilon}{2}
$$

Therefore using (A.2) and (A.7) we obtain

$$
|y(t) - \hat{y}(t)| \leq \frac{\epsilon}{2} + l_h |e_x(t)| \leq \epsilon \qquad \forall t \in [0, T]
$$

which concludes the proof. □

APPENDIX B : Proof of Lemma 1

We will proof here only part (i) since the proof of part (ii) follows from the same reasoning. Suppose $\|W_1\|_F = M_1$ and $e_x^T W_1 \xi < 0$. If that is not the case then $I_1^* = 0$ and the inequality holds trivially. The term $\mathrm{tr}\left\{W_1 \Phi_1^T\right\}$ can be expressed as

$$
\begin{aligned}
\mathrm{tr}\left\{W_1 \Phi_1^T\right\} &= \mathrm{tr}\left\{(\Phi_1 + W_1^*)\Phi^T\right\} \\
&= \mathrm{tr}\left\{\frac{1}{2}\Phi_1\Phi_1^T + \left(\frac{1}{2}\Phi_1\Phi_1^T + W_1^*\Phi^T\right)\right\} \\
&= \frac{1}{2}\|\Phi_1\|_F^2 + \frac{1}{2}\|W_1\|_F^2 - \frac{1}{2}\|W_1^*\|_F^2
\end{aligned}
$$

Therefore

$$
I_1^* \frac{e_x^T W_1 \xi}{\|W_1\|_F^2} \mathrm{tr}\left\{W_1 \Phi_1^T\right\} = \frac{1}{2M_1^2}\left(e_x^T W_1 \xi\right)\left(\|\Phi_1\|_F^2 + M_1^2 - \|W_1^*\|_F^2\right) \tag{B.1}
$$

Since the optimal weights W_1^* satisfy $\|W_1^*\|_F \leq M_1$, the last term in (B.1) is positive and therefore

$$I_1^* \frac{e_x^T W_1 \xi}{\|W_1\|_F^2} \operatorname{tr}\left\{W_1 \Phi_1^T\right\} \leq 0$$

$\square$

References

[1] G.C. Goodwin and K.S. Sin, *Adaptive Filtering Prediction and Control*, Englewood Cliffs, NJ, Prentice-Hall, 1984.

[2] K.S. Narendra and A.M. Annaswamy, *Stable Adaptive Systems*, Englewood Cliffs, NJ, Prentice-Hall, 1989.

[3] S. Sastry and M. Bodson, *Adaptive Control: Stability, Convergence and Robustness*, Englewood Cliffs, NJ, Prentice-Hall, 1989.

[4] K.J. Aström and B. Wittenmark, *Adaptive Control*, Reading, MA, Addison-Wesley, 1989.

[5] A. Isidori, *Nonlinear Control Systems*, Berlin, Springer-Verlag, 1989.

[6] H. Nijmeijer and A.J. van der Schaft, *Nonlinear Dynamical Control Systems*, New York, NY, Springer-Verlag, 1990.

[7] D.G. Taylor, P.V. Kokotovic, R. Marino, and I. Kanellakopoulos, "Adaptive regulation of nonlinear systems with unmodelled dynamics", *IEEE Trans. Aut. Control*, vol. AC-34, pp. 405-412, April 1989.

[8] S.S. Sastry and A. Isidori, "Adaptive control of linearizable systems", *IEEE Trans. Aut. Control*, vol. AC-34, pp. 1123-1131, November 1989.

[9] L. Praly, G. Bastin, J.-B. Pomet and Z.P. Jiang, "Adaptive Stabilization of Nonlinear Systems", in *Foundations of Adaptive Control*, P.V. Kokotovic ed., pp. 347-433, Springer-Verlag, Berlin, 1991.

[10] P.V. Kokotovic, I. Kanellakopoulos, and A.S. Morse, "Adaptive Feedback Linearization of Nonlinear Systems", in *Foundations of Adaptive Control*, P.V. Kokotovic ed., pp. 311-346, Springer-Verlag, Berlin, 1991.

[11] T.W. Miller, R.S. Sutton III, and P.J. Werbos, Eds, *Neural Networks for Control*, Cambridge, MA, The MIT Press, 1990.

[12] P.J. Antsaklis, Editor, Special Issue on Neural Networks in Control Systems, *IEEE Control Systems Magazine*, vol. 10, No.3, April 1990.

[13] K.S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks", *IEEE Trans. on Neural Networks*, vol. 1, pp.4-27, 1990.

[14] M.I. Jordan and D.E. Rumelhart, "Forward models: supervised learning with a distal teacher", *Occ. Paper #40, Center for Cognitive Science*, M.I.T., 1990.

[15] M.S. Lan, "Adaptive control of unknown dynamical systems via neural network approach", *Proc. Automatic Control Conf.*, pp. 910-915, 1989.

[16] F.C. Chen and H.K. Khalil, "Adaptive control of nonlinear systems using neural networks", *Proc. IEEE Conf. on Decision and Control*, pp. 1707-1712, 1990.

[17] B.E. Ydstie, "Forecasting and control using adaptive connectionist networks", *Computers chem. Engng.*, vol. 14, pp. 583-599, 1990.

[18] R.J. Williams and D. Zipser, "A learning algorithm for continuously running fully recurrent neural networks", *Neural Computation*, vol. 1, pp. 270-280, 1989.

[19] P. J. Werbos, "Backpropagation through time: what it does and how to do it", *Proc. IEEE*, vol. 78, pp. 1550-1560, 1990.

[20] K.S. Narendra and K. Parthasarathy, "Gradient methods for the optimization of dynamical systems containing neural networks", *IEEE Trans. on Neural Networks*, vol. 2, pp. 252-262, 1991.

[21] J.B. Cruz., Jr., Ed., *System Sensitivity Analysis*, (Benchmark papers in Electrical Engineering and Computer Science), Stroudsburg, PA, Dowden, Hutchinson and Ross, 1973.

[22] H.P. Whitaker, J. Yamron, and A. Kezer, "Design of model-reference adaptive control systems for aircraft", *Report R-164, Instrumentation Lab., MIT*, 1958.

[23] P.C. Parks, "Lyapunov redesign of model reference adaptive control systems", *IEEE Trans. Aut. Control*, vol. AC-11, pp. 362-367, 1966.

[24] D.J. James, "Stability of a model reference control system", *AIAA Journal*, vol. 9, no. 5, 1971.

[25] I.M.Y. Mareels, B.D.O. Anderson, R.R. Bitmead, M. Bodson, and S.S. Sastry, "Revisiting the MIT rule for adaptive control", *Proc. of the 2nd IFAC Workshop on Adaptive Systems in Control and Signal Processing*, Lund, Sweden, 1986.

[26] A.G. Barto, "Connectionist learning for control: an overview", in *Neural Networks for Control*, T.W. Miller, R.S. Sutton III, and P.J. Werbos, Eds, pp. 5-58, Cambridge, MA, The MIT Press, 1990.

[27] M.M. Polycarpou and P.A. Ioannou, "Learning and convergence analysis of neural-type structured networks", to appear in *IEEE Trans. on Neural Networks*, vol. 3, no. 1, 1992.

[28] R.M. Sanner and J.-J.E. Slotine, "Guassian networks for direct adaprive control", *Proc. Autom. Control Conf.*, pp. 2153-2159, 1991.

[29] A. Arapostathis, B. Jakubczyk, H.G. Lee, S.I. Marcus, and E.D. Sontag, "The effect of sampling on linear equivalence and feedback linearization", *Systems & Control Letters*, vol. 13, pp. 373-381, 1989.

[30] D.E. Rumelhart, J.L. McClelland and the PDP Research group, *Parallel Distributed Processing: Exploration in the Microstructure of Cognition. Volume 1: Foundations*, Cambridge, MA, The MIT Press, 1986.

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

[32] J. Moody and C.J. Darken, "Fast learning in networks of locally-tuned processing units", *Neural Computation*, vol. 1, pp. 281-294, 1989.

[33] S. Renals and R. Rohwer, "Phoneme classification experiments using radial basis functions", *Proc. Intern. Joint Conf. on Neural Networks*, vol. 1, pp. 461-467, 1989.

[34] M. Niranjan and F. Fallside, "Neural networks and radial basis functions for classifying static speech patterns", *Computer Speech and Language*, vol. 4, pp. 275-289, 1990.

[35] T. Poggio and F. Girosi "A theory of networks for approximation and learning", *Tech. Rep. Artif. Intel. Lab, Memo No. 1140*, M.I.T., 1989.

[36] T. Poggio and F. Girosi "Extensions of a theory of networks for approximation and learning: dimensionality reduction and clustering", *Tech. Rep. Artif. Intel. Lab, Memo No. 1167*, M.I.T., 1990.

[37] T. Poggio and F. Girosi "Regularization algorithms for learning that are equivalent to multilayer networks", *Science*, vol. 247, pp. 978-982, 1990.

[38] G. Cybenko, "Approximation by superpositions of a sigmoidal function", *Mathematics of Control, Signals, and Systems*, vol. 2, pp. 303-314, 1989.

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

[40] E.J. Hartman, J.D. Keeler, and J.M. Kowalski, "Layered neural networks with guassian hidden units as universal approximations", *Neural Computation*, vol. 2, pp. 210-215, 1990.

[41] K. Funahashi, "On the approximate realization of continuous mappings by neural networks", *Neural Networks*, vol. 2, pp.183-192, 1989.

[42] E.D. Sontag, "Feedback stabilization using two-hidden-layer nets", *Proc. Autom. Control Conf.* pp. 815-820, 1991.

[43] D.S. Broomhead and D. Lowe, "Multivariable functional interpolation and adaptive networks", *Complex Systems*, vol. 2, pp. 321-355, 1988.

[44] J. Park and I.W. Sandberg, "Universal approximation using radial-basis-function networks", *Neural Computation*, vol. 3, pp. 246-257, 1990.

[45] J.K. Hale, *Ordinary Differential Equations*, New York, NY, Wiley-InterScience, 1969.

[46] G.H. Golub and C.F. Van Loan, *Matrix Computations*, The John Hopkins Univ. Press, 2nd Edition, 1989.

[47] T. Holcomb and M. Morari, "Local training of radial basis function networks: towards solving the hidden unit problem", *Proc. Autom. Control Conf.*, pp. 2331-2336, 1991.

[48] P. Kudva and K.S. Narendra, "Synthesis of an adaptive observer using Lyapunov's direct method", *Inter. Journal of Control*, vol. 18, pp. 1201-1210, 1973.

[49] P.A. Ioannou and P.V. Kokotovic, *Adaptive Systems with Reduced Models*, Springer-Verlag, New York, NY, 1983.

[50] G.C. Goodwin and D.Q. Mayne, "A parameter estimation perspective of continuous time model reference adaptive control", *Automatica*, vol. 23, pp. 57-70, Jan. 1987.

[51] M.M. Polycarpou and P.A. Ioannou, "On the existence and uniqueness of solutions in adaptive control systems", *Tech. Rep. No. 90-05-01, Dept. Elec. Eng. - Systems, Univ. of Southern Cal.*, 1990.

[52] P.A. Ioannou and K.S. Tsakalis, "A robust direct adaptive controller", *IEEE Trans. Aut. Control*, vol. AC-31, pp. 1033-1043, Nov. 1986.

[53] K.S. Narendra and A.M. Annaswamy, "A new adaptive law for robust adaptation with persistent excitation", *IEEE Trans. on Aut. Control*, vol. AC-32, no. 2, pp. 134-145, 1987.

[54] P.A. Ioannou and A. Datta, "Robust Adaptive Control: Design, Analysis and Robustness Bounds", in *Foundations of Adaptive Control*, P.V. Kokotovic ed., pp. 71-152, Springer-Verlag, Berlin, 1991.

[55] D.G. Luenberger, *Linear and Nonlinear programming*, Addison-Wesley Publ. Co., 1984.

[56] P.V. Kokotovic, I. Kanellakopoulos, "Adaptive nonlinear control: a critical appraisal", *Proc. 6th Yale Workshop on Adaptive and Learning Systems*, New Haven, CT, 1990.

[57] T. Kailath, *Linear Systems*, Englewood Cliffs, NJ, Prentice-Hall, 1980.

[58] P.A. Ioannou, F. Giri, and F. Ahmed-Zaid, "Stable indirect adaptive control: the switched-excitation approach", *Tech. Rep. No. 91-05-02, Dept. Elec. Eng. - Systems, Univ. of Southern Cal.*, 1991.

[59] V.I. Utkin, *Sliding Modes and their Application to Variable Structure Systems*, Moscow, MIR Publishers, 1978.

[60] J.-J.E. Slotine and W. Li, *Applied Nonlinear Control*, Englewood Cliffs, NJ, Prentice-Hall, 1991.

[61] K.D. Young, P.V. Kokotovic, and V.I. Utkin, "A singular perturbation analysis of high-gain feedback systems", *IEEE Trans. on Autom. Control*, vol. AC-22, no. 6, pp. 931-938, 1977.