A Direct Control Method For a Class of Nonlinear Systems Using Neural Networks

E. Tzirkel-Hancock & F. Fallside

CUED/F-INFENG/TR.65

Cambridge University Engineering Department
Trumpington Street
Cambridge CB2 1PZ
England

March 1991

email: et@uk.ac.cam.eng

The first author is supported by grants from Trinity College Cambridge and the Committee of Vice-Chancellors and Principals

Abstract

A direct control scheme for a class of continuous time nonlinear systems using neural networks is presented. The objective of control is to track a desired reference signal. This objective is achieved through input/output linearization of the system with neural networks. The network learning is based on a stability type algorithm in which learning and control take place simultaneously. The method is analysed in light of its close relation to adaptive control methods. This analysis provides an interesting bridge between well studied, rigorous adaptive control methods and the field of neural network training. In particular, the importance of the property of *persistent excitation* and its implications to learning with networks of localized receptive fields is discussed.

1 Introduction

The class of systems we wish to control is defined by the differential equations

$$\begin{array}{rcl}\dot{x} & = & f(x) + g_1(x)u_1 + \cdots g_l(x)u_l \\ y_1 & = & h_1(x) \\ & \cdot & \\ & \cdot & \\ & \cdot & \\ y_l & = & h_l(x)\end{array} \tag{1}$$

where $f : \Re^n \to \Re^n$, $g_i : \Re^n \to \Re^n$ $1 \leq i \leq l$ and $h_i : \Re^n \to \Re$ $1 \leq i \leq l$ are smooth functions on the state space $x \in \Re^n$, and u_i, y_i $1 \leq i \leq l$ are the system's inputs and outputs respectively. This type of system is encountered in many applications, e.g. rigid link robot manipulator control.

A variety of methods for control of this type of system in the case where the functions f, g_i, h_i $1 \leq i \leq l$ are known, have been developed by nonlinear control theorists. A good survey of these methods is available in [4]. Adaptive control methods for linearization and control of this system have also been proposed [12, 13]. These methods are based on the assumption that all of the system's nonlinear functions can be expressed as linear combinations of known functions.

In what follows a direct control method for this system, using neural networks (NN) is presented. The method is close in nature to the adaptive control approach. It differs in that it does not assume any knowledge of plant nonlinearities. For the simplicity of presentation we will first discuss the single input single output (SISO) case, i.e. when $l = 1$. In section 5 the method will be extended to MIMO systems.

The system equations in the SISO case are simplified to

$$\begin{array}{l}\dot{x} = f(x) + g(x)u \\ y = h(x)\end{array} \tag{2}$$

The control method is based on input/output linearization of this system with static state feedback, where the feedback is generated by two neural networks. The networks' role is to directly approximate the Lie derivatives of the function h along the vector fields f and g. The ability of the networks to accurately approximate these derivatives is a key requirement for the control method to work.

The neural networks consist of two layers. The first layer consists of receptive fields distributed over the relevant state space. We employ radial basis functions (RBF) as receptive fields [10]. The second layer performs a simple weighted summation of the first layer outputs. The second layer parameters are the only adjustable parameters and their adaptation is based on a stability type algorithm. The algorithm behaviour is studied with the aid of adaptive control principles.

In section 2, a brief review of input/output linearization and related concepts is given. Section 3 introduces the control law, the network adaptation law and analyses the properties of the control method. In section 4, the results of two simulations are presented and in section 5 the method is extended to control MIMO systems.

2 On Exact Linearization

In this section we briefly review a technique for exact input/output linearization of the plant (2) and some related concepts. For a detailed treatment see [4, 6].

2.1 Strong Relative Degree and Exact Linearization

Differentiating the plant output y with respect to time we get

$$\dot{y} = L_f h(x) + L_g h(x) u \tag{3}$$

where $L_f h(x)$ is the Lie derivative of the function h along the vector field f i.e. $L_f h(x) = \frac{\partial h(x)}{\partial x} f(x)$. If $L_g h(x) = 0$ then $\dot{y}$ does not depend directly on the input and we differentiate again to get

$$\ddot{y} = L_f^2 h(x) + L_g L_f h(x) u \tag{4}$$

where $L_f^2 h(x) = L_f((L_f h)(x))$ and $L_g L_f h(x) = L_g((L_f h)(x))$. If the second derivative is still not directly dependent on the input, we can keep differentiating in this way until some derivative of y depends directly on the input. This process leads to the following definition of *strong relative degree.*

Definition 1 *The system (2) is said to have* strong relative degree γ *if* $L_g h(x) = L_g L_f h(x) = \cdots = L_g L_f^{\gamma-2} h(x) = 0$ *for all* x *and* $L_g L_f^{\gamma-1} h(x)$ *is bounded away from zero for all* x.

If a system has a strong relative degree γ then $\gamma \leq n$. Also, by differentiating the output as above one can verify that the first $\gamma - 1$ time derivatives of y do not directly depend on the input and the γ-order derivative is given by

$$y^{(\gamma)} = L_f^{\gamma} h(x) + L_g L_f^{\gamma-1} h(x) u \tag{5}$$

If the plant (2) has strong relative degree γ then a static state feedback of the form

$$u = \frac{1}{L_g L_f^{\gamma-1} h(x)} (v - L_f^{\gamma} h(x)) \tag{6}$$

yields the linearized equation

$$y^{(\gamma)} = v \tag{7}$$

The linearizing feedback law (6) is the basis for our control method, in which the exact Lie derivatives are replaced with their neural network approximations.

2.2 Minimum Phase Systems

It can be shown that if the plant (2) has a strong relative degree γ, then a state transformation $T : \Re^n \to \Re^n$ exists such that the transformed states $z = T(x)$ satisfy the following differential equation, called the *normal form* [6].

$$\begin{aligned}
\dot{z}_1 &= z_2 \\
&\vdots \\
\dot{z}_{\gamma-1} &= z_\gamma \\
\dot{z}_\gamma &= f_1(z) + g_1(z)u = f(x) + g(x)u \\
\dot{z}_{\gamma+1} &= K_{\gamma+1}(z) \\
&\vdots \\
\dot{z}_n &= K_n(z) \\
y &= z_1
\end{aligned} \tag{8}$$

The transformation can be selected in such a way that the dynamics of the states $z_{\gamma+1} \ldots z_n$ do not depend directly on the input. The dynamics of these states, when the output is forced to be zero, are called the system's *zero dynamics*. The plant is called *minimum phase* if the zero dynamics are asymptotically stable. A minimum phase assumption is required if the plant is to be controlled in a stable manner by means of exact input/output linearization [1].

3 Direct Control with Neural Networks

3.1 Assumptions and Control Law

We make the following assumptions with regard to the plant (2)

Assumption 1 *The plant has a known strong relative degree γ, is globally exponentially minimum phase, the sign of $L_g L_f^{\gamma-1} h(x)$ is known and a lower bound of $|L_g L_f^{\gamma-1} h(x)|$ is known.*

Assumption 2 *The Lie derivatives $L_f^\gamma h(x)$ and $L_g L_f^{\gamma-1} h(x)$ are sufficiently regular that they can be uniformly approximated by neural networks of the form described by*

[1] Conditions under which the normal form and the zero dynamics can be defined globally in the state space are discussed in [12]

the product $\theta^T w(x)$, where $w(x)$ is a layer of receptive fields defined over the state space and θ is an adjustable parameter vector.

Assumption 3 *The state x of the plant and the output derivatives up to order $\gamma - 1$ are available for measurement.*

Assumption (2) needs to be further clarified. The Lie derivatives $L_f^\gamma h(x)$ and $L_g L_f^{\gamma-1} h(x)$ are approximated by the networks $\theta_f^T w_f(x)$ and $\theta_g^T w_g(x)$ respectively where the vectors $w_f(x)$, $w_g(x) : \Re^n \rightarrow \Re^p$ contain the basis functions of the networks and the vectors θ_f, $\theta_g \in \Re^p$ are the networks' parameter vectors [2]. Assumption (2) states that it is possible to find basis function vectors $w_f(x)$ and $w_g(x)$ so that the error bounds defined by the following equation are acceptably small.

$$\begin{array}{rcl} \bar{f} & = & \min_{\theta_f} \max_{x \in X} |L_f^\gamma h(x) - \theta_f^T w_f(x)| \\ \bar{g} & = & \min_{\theta_g} \max_{x \in X} |L_g L_f^{\gamma-1} h(x) - \theta_g^T w_g(x)| \end{array} \quad (9)$$

where $X \subseteq \Re^n$ stands for the relevant state space.

Using this definition of the bounds $\bar{f}$ and $\bar{g}$, one can find *optimal* parameter vectors $\breve{\theta}_f$ and $\breve{\theta}_g$ such that

$$\text{where:} \quad \begin{array}{rcl} L_f^\gamma h(x) & = & \breve{\theta}_f^T w_f(x) + d_f(x) \\ L_g L_f^{\gamma-1} h(x) & = & \breve{\theta}_g^T w_g(x) + d_g(x) \\ & & \\ |\, d_f(x) \,| & \leq & \bar{f} \quad \text{for all } x \in X \\ |\, d_g(x) \,| & \leq & \bar{g} \quad \text{for all } x \in X \end{array} \quad (10)$$

Here we choose radial basis function (RBF) for the approximating networks since they are known to be suitable for function approximation and are widely used for this purpose [10, 11].

Assumption (3) is necessary since the states and output derivatives are needed for feedback generation. The requirement of output derivative availability restricts the class of systems to which this control method may be applied. Nevertheless, the output derivatives are available as states in a large class of electromechanical systems, which include the rigid link robot manipulator as a special case.

Given assumptions (1), (2) and (3) we define the control signal

$$u = \frac{v - \theta_f^T w_f(x)}{\theta_g^T w_g(x)} \quad (11)$$

which is similar in structure to the control signal defined in equation (6). The adaptive law of the parameter vector θ_g must ensure that $\theta_g^T w_g(x)$ is bounded away from zero at all times; this is where the lower bound from assumption (1) is needed.

---

[2] For notational simplicity it has been assumed that both networks have the same number of basis functions

We now define the parameter error vectors

$$\begin{aligned}\phi_f &= \theta_f - \breve{\theta}_f \\ \phi_g &= \theta_g - \breve{\theta}_g\end{aligned} \tag{12}$$

and the vectors $\phi^T = [\phi_f^T \;\phi_g^T]$, $W(x,u)^T = [-w_f(x)^T \;\; - uw_g(x)^T]$ and $d(x,u) = d_f(x) + ud_g(x)$. With these definitions and by inserting the control law (11) to the plant (2) we get the input/output relation

$$\begin{aligned}y^{(\gamma)} &= -\phi_f^T w_f(x) + d_f(x) + u[-\phi_g^T w_g(x) + d_g(x)] + v \\ &= \phi^T W(x,u) + d(x,u) + v\end{aligned} \tag{13}$$

This equation is the counterpart of equation (7), except that here additional forcing terms that arise from the parameter error and inherent approximation errors appear on the right hand side.

With the objective of tracking the reference signal y_m we now define the tracking error $e = y - y_m$ and use a linear feedback law

$$v = y_m^{(\gamma)} - \alpha_\gamma e^{(\gamma-1)} - \cdots - \alpha_1 e \tag{14}$$

where the polynomial $\Gamma(s) = s^\gamma + \alpha_\gamma s^{\gamma-1} + \cdots + \alpha_1$ is Hurwits, i.e. has all its roots in the open left half of the complex plane. Note that availability of the output derivatives is required here.

With the control v inserted into equation (13) we get a differential equation for the tracking error

$$e^{(\gamma)} + \alpha_\gamma e^{(\gamma-1)} + \cdots + \alpha_1 e = \phi^T W(x,u) + d(x,u) \tag{15}$$

For the sake of notation compactness we take the liberty of mixing time and frequency domain notations and write this error equation in the form

$$e = \Gamma^{-1}\{\phi^T W(x,u) + d(x,u)\} \tag{16}$$

This type of error equation is frequently encountered in adaptive control. In those instances, the vector W is called the regressor vector and d is usually a disturbance due to measurement noise or unmodelled dynamics. Here the vector $W(x,u)$ consists of a collection of the two neural networks' basis functions and $d(x,u)$ is a disturbance due to the network inherent approximation inaccuracies.

3.2 The Adaptive Law

We now present the networks' adaptive law which is derived in similar fashion as it is done with an output error adaptive control scheme [9, 12].

The error e from equation (16) is modified as follows

$$e_1 = \beta_\gamma e^{(\gamma-1)} + \cdots + \beta_1 e \stackrel{\triangle}{=} \Psi(s)\, e \tag{17}$$

where $\Psi(s)$ is selected in such a way that the transfer function $\Omega(s) = \frac{\Psi(s)}{\Gamma(s)}$ is *strictly positive real* (SPR) [9, 12]. The modified error is then given by the equation

$$e_1 = \Omega(s)\{\phi^T W(x,u) + d(x,u)\} \tag{18}$$

where we have again mixed frequency and time domain notations for convenience. This modified error is used in the parameter update law

$$\dot{\theta} = \dot{\phi} = -\eta e_1 W(x,u) \tag{19}$$

were $\eta > 0$ is a learning rate.

This parameter adaptation law, reminiscent of a gradient rule is based on a stability type argument. We proceed by introducing the Lyapunov function that corresponds to the joint adaptive process given by the error equation (18) and the update law (19).

First we note that as the transfer function $\Omega(s)$ is SPR, the error e_1 can be expressed as the output of the linear system

$$\begin{aligned} \dot{\epsilon} &= A\epsilon + b\{\phi^T W(x,u) + d(x,u)\} \\ e_1 &= c^T \epsilon \end{aligned} \tag{20}$$

where $\epsilon, b, c \in \Re^\gamma$, $A \in \Re^{\gamma\times\gamma}$ and A, b and c satisfy

$$\begin{aligned} PA + A^T P &= -Q \\ Pb &= c \end{aligned} \tag{21}$$

where $P \in \Re^{\gamma\times\gamma}$ and $Q \in \Re^{\gamma\times\gamma}$ are symmetric positive definite matrices [2]. Using this representation we define the Lyapunov function

$$V = \eta \epsilon^T P \epsilon + \phi^T \phi \tag{22}$$

Taking the time derivative of this function along the trajectories of the error equation (20) and the update law (19) we get after some manipulations

$$\dot{V} = \eta\{-\epsilon^T Q \epsilon + 2\epsilon^T c d(x,u)\} \tag{23}$$

Suppose now that the networks map the Lie derivatives perfectly, i.e $d(x,u) \equiv 0$. It then follows from equation (23) that V is decreasing with time along the system's trajectory. From this we may conclude that the tracking error converges asymptotically to zero and that all the system's signals and parameters are bounded. Additionally, the parameter error vector converges exponentially to zero if the regressor vector $W(x,u)$ is *persistently excited* [9, 12]. Analysis of the disturbance $d(x,u)$ on the system behaviour is closely related to robustness analysis in adaptive control, the main difference being the source of the disturbance.

3.3 Analysis

In this section we discuss the issues of parameter convergence and the overall system behaviour with the inherent approximation limitations.

From equation (23) it can be concluded that $\dot{V}$ is negative as long as

$$\| \epsilon \| \geq \frac{2 \| c \|}{\lambda_{\min} Q} \bar{d} \triangleq \bar{\epsilon} \tag{24}$$

where $\lambda_{\min} Q$ is the smallest eigenvalue of matrix Q, and $\bar{d}$ is an upper bound on $d(x, u)$. If $\bar{u}$ is an upper bound on the control u then $\bar{d}$ in turn satisfies

$$\bar{d} \leq \bar{f} + \bar{u}\bar{g} \tag{25}$$

The bound expressed in equation (24) is significant because it quantifies the connection between the networks approximation error bounds and the tracking error. In particular, if we stop updating the parameters whenever the error is smaller than this bound then the error can be shown to be asymptotically bounded by $\bar{\epsilon}$ [3] [12].

The error bound $\bar{\epsilon}$ can be reduced by refining our networks to reduce the approximation bounds $\bar{f}$ and $\bar{g}$. We also have full control over the transfer function $\Omega(s)$. This can be used to reduce the tracking error bound, although we should keep in mind that $\Omega(s)$ influences other properties of the overall system; in particular, large feedback can result in unacceptable chatter and large control activity. The importance of the bound $\bar{\epsilon}$ is due to its existence and is mainly of theoretical value. In practice it is usually impossible to compute this bound. Note that without using the deadzone method the error is not guaranteed to be asymptotically bounded by $\bar{\epsilon}$. Simulations however suggest that the tracking error can normally be reduced to values substantially smaller then $\bar{\epsilon}$.

Parameter convergence and robustness of the method to inherent approximation errors are issues strongly related to the property known in adaptive control as *persistent excitation* (PE). A vector $w(t)$ is PE if there exist α_1, α_2, $\delta > 0$ such that

$$\alpha_2 I \geq \int_t^{t+\delta} w(\tau) w^T(\tau) d\tau \geq \alpha_1 I \qquad \text{for all } t \geq 0 \tag{26}$$

where I is the identity matrix. The constant α_1 is called *the degree of excitation.* A physical interpretation of condition (26) is that a persistently excited vector $w(t)$ has energy in every direction.

The following facts are known in adaptive control theory [9, 12]:

1. The unperturbed dynamic system defined by the SPR error equation (18) with a *zero disturbance,* $d \equiv 0$, and the update law (19) is exponentially stable if the vector W is PE. That is, both the tracking error and the parameter error vector converge to zero in an exponential manner. If the learning rate is small then the rate of convergence is approximately proportional to $\eta \frac{\alpha_1}{\delta}$. Thus PE guarantees exponential convergence and an increase in the degree of PE increases the convergence rate.

[3] This corresponds to the so called deadzone method in adaptive control

2. The perturbed system defined by the SPR error equation (18), with the disturbance d, and the update law (19), has a certain degree of robustness. That is, there exist positive constants c and l such that if $\| d(x,u) \|_\infty \leq c$ [4] then the joint error vector $[e_1 \ \phi^T]^T$ converges to a ball of radius $l \ \| d(x,u) \|_\infty$ centred at the origin. Moreover, the so called *stability margin* constant c can be increased and the *gain* l decreased if the rate of exponential convergence of the system is increased. For more details see [12] pp. 221-231.

From these points it follows that a sufficient degree of PE can, under some conditions, guarantee asymptotically bounded tracking error and parameters error. An increase in the degree of excitation increases the bound on tolerable disturbance and reduces the asymptotic error bounds.

Unfortunately it is impossible to guarantee persistent excitation or the degree of excitation for our control scheme. We will limit ourselves to a qualitative discussion of how the PE property is projected to network learning in the case when the network consists of local receptive fields.

The vector $W(x,u)$ was previously defined as $W(x,u) = [-w_f^T(x) \ \ -uw_g^T(x)]^T$. We will first consider the persistent excitation of the vector $w_f(x)$. Suppose that $w_f(x)$ consists of localized receptive fields that fully cover the relevant state space and with no overlap between the fields. Then the receptive fields are orthogonal and the integral in equation (26) of the off-diagonal elements of $w_f(x)w_f^T(x)$ vanish. Thus for PE to hold, all receptive fields must be activated within every interval of time of length δ. If there is some (small) overlap between the receptive fields, $w_f(x)$ can still be persistently excited if all receptive fields are strongly activated within every interval of time of length δ, in which case, the symmetric matrix defined by the integral in equation (26) is strictly diagonally dominant and therefore positive definite [5]. It follows that for PE to hold, the input should be sufficiently rich so that the relevant state space will be fully searched within every time interval of length δ, i.e. fully searched in the sense that all receptive fields are strongly activated. The PE property is thus analogous in this case to the typical requirement, in neural network training, of the training set to be rich and to be presented fully and frequently

The persistent excitation analysis of the vector $W(x,u)$ is far more difficult. This is due to the presence of the control u in $W(x,u)$ and the fact that the local receptive fields in $w_f(x)$ cannot be orthogonal, or nearly orthogonal, to the receptive fields of $w_g(x)$. Nevertheless, experimentation suggests that with a rich reference, in the sense described above, good approximation of the Lie derivatives over the desired state space region can be achieved. Experiments also suggest that with local receptive fields, parameter divergence away from their desirable values or to infinity is unlikely. The reason for this may be that only parameters of receptive fields that are active at any one time are updated at that time and the rest remain unchanged.

The need for a rich reference to achieve good approximation means that in practice it is desirable to incorporate an initial control/identification phase of the plant with such a rich input, before engaging in operation with the actual reference signals. It was found experimentally that adaptation can proceed in actual operation, when

[4] The symbol $\| \cdot \|_\infty$ stands for the infinity norm with respect to time

the reference is not rich; this usually results in improved approximations in state regions that are part of the trajectory with other state space regions remaining mostly unaffected. One should keep in mind however that in some circumstances it may be required to switch adaptation off, in which case there will be two distinct phases, one of control/identification and another of control only, without adaptation.

To complete this discussion it is important to stress that the feasibility of the proposed control method is strongly dependent on the networks' ability to closely approximate the Lie derivatives. The choice of suitable networks remains largely an experimental task. Initial information about the plant nonlinearities can help with this task.

4 Simulations

4.1 Simulation 1

We consider the control of a single state plant of the type (2) with

$$\begin{aligned} f(x) &= \sin(x) + 0.5\cos(3x) \\ g(x) &= 1 + 4x^2 \\ h(x) &= 2x + 0.5\sin(2x). \end{aligned} \tag{27}$$

For this plant we get $L_g h(x) = (2 + \cos(2x))(1 + 4x^2)$ which is bounded away from zero for all x.

We use two similar RBF networks for the approximation. Each network contains 41 gaussian functions, each with standard deviation 0.06, spread uniformly on the state interval $x \in [-1, 1]$. The control and adaptive laws are those of equations (11) (14) and (19), with $v = \dot{y}_m - \alpha(y - y_m)$ where $\alpha = 1$ and where y_m is the output of the reference model $\dot{y}_m = -y_m + r$ and r is the reference input [5]. We choose $e_1 = e$ since the transfer function $\frac{1}{s+1}$ is already SPR. We experimentally find the Lie derivatives approximation error bounds over the relevant state interval to be $\bar{f} = 0.006$ and $\bar{g} = 0.025$ [6].

Two experiments were carried out with this system. The first included an initial control/identification phase with a rich reference signal, and the second did not. The results of the first experiment are depicted in figures 1 and 2 and the results of the second in figures 3, 4 and 5

Figure 1 depicts the approximation of the Lie derivatives after an initial training phase with a band limited noise with a cutoff frequency $w_c = 4$ as the reference r. The learning rate was $\eta = 5$ and training took about 33 minutes of system real time (4 minutes of sun4 cpu time). We can clearly see that in the region $x \in [-0.6, 0.6]$ in which most of the state signal is contained, the approximations are good.

The controller performance after the initial identification phase, with a square wave reference input of magnitude 1, is depicted in figure 2. Adaptation was switched off. The model reference and the plant outputs (top) are seen to be indistinguishable.

[5] In the simulations reported here we consider model reference adaptive control tasks

[6] These bounds were found through offline network training

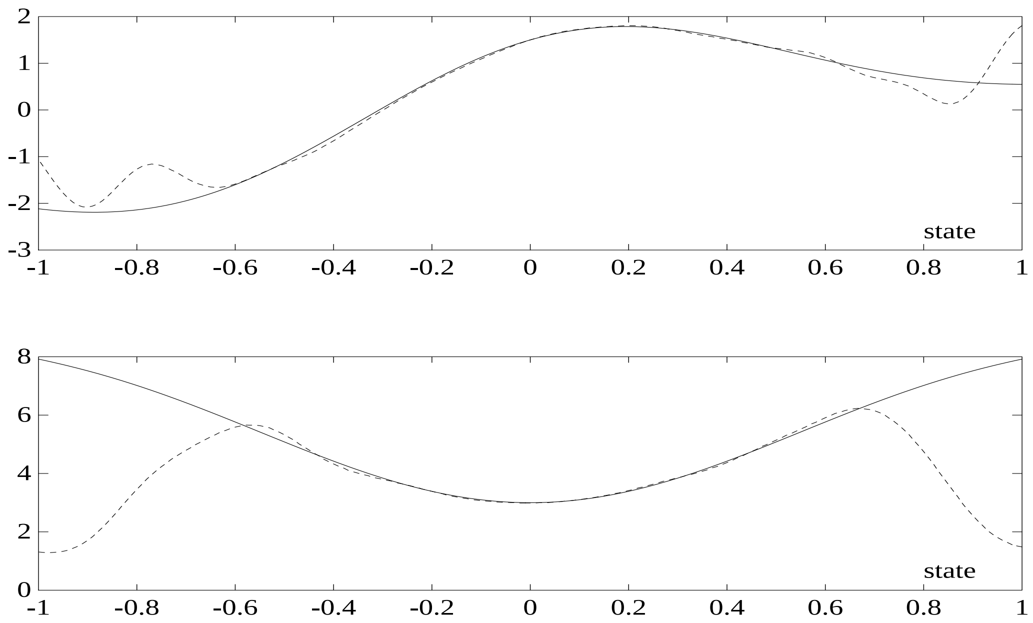

Figure 1: $L_f h(x)$ (top) and $L_g h(x)$ (bottom) approximations after an identification phase. The solid lines are the derivatives and the dashed lines their approximations.

The middle and bottom parts of the figure shows the tracking error and control respectively. We note that with a control bound of 1.5 and the Lie derivatives approximation bounds given above, the error bound in equation (24) is $\bar{d}/\alpha = 0.044$. The actual tracking error is forty times smaller than this bound!

We carried out a further 33 minutes of control with the square reference input and an adaptation gain of $\eta = 1$. We found that even though adaptation was not switched off, the approximation accuracy did not deteriorate. In fact it was improved in some regions of the state space.

In the second experiment we tried to control the plant with the same square wave reference but without an initial identification phase. The learning rate was $\eta = 5$ and learning time was 33 minutes. The tracking error during the first 3 minutes of learning is depicted in figure 3. Figure 4 depicts the tracking error after 33 minutes has elapsed. Tracking is seen to be good although the error is ten times larger than with the initial identification phase.

Figure 5 shows the Lie derivatives approximation errors after 33 minutes of control with the square wave. The errors are seen to be much larger than those depicted in figure 1. This is consistent with the lower degree of excitation in this case. The large approximation errors may result in a large tracking error when the reference changes its nature. Thus it is desirable to precede control with an identification phase as above.

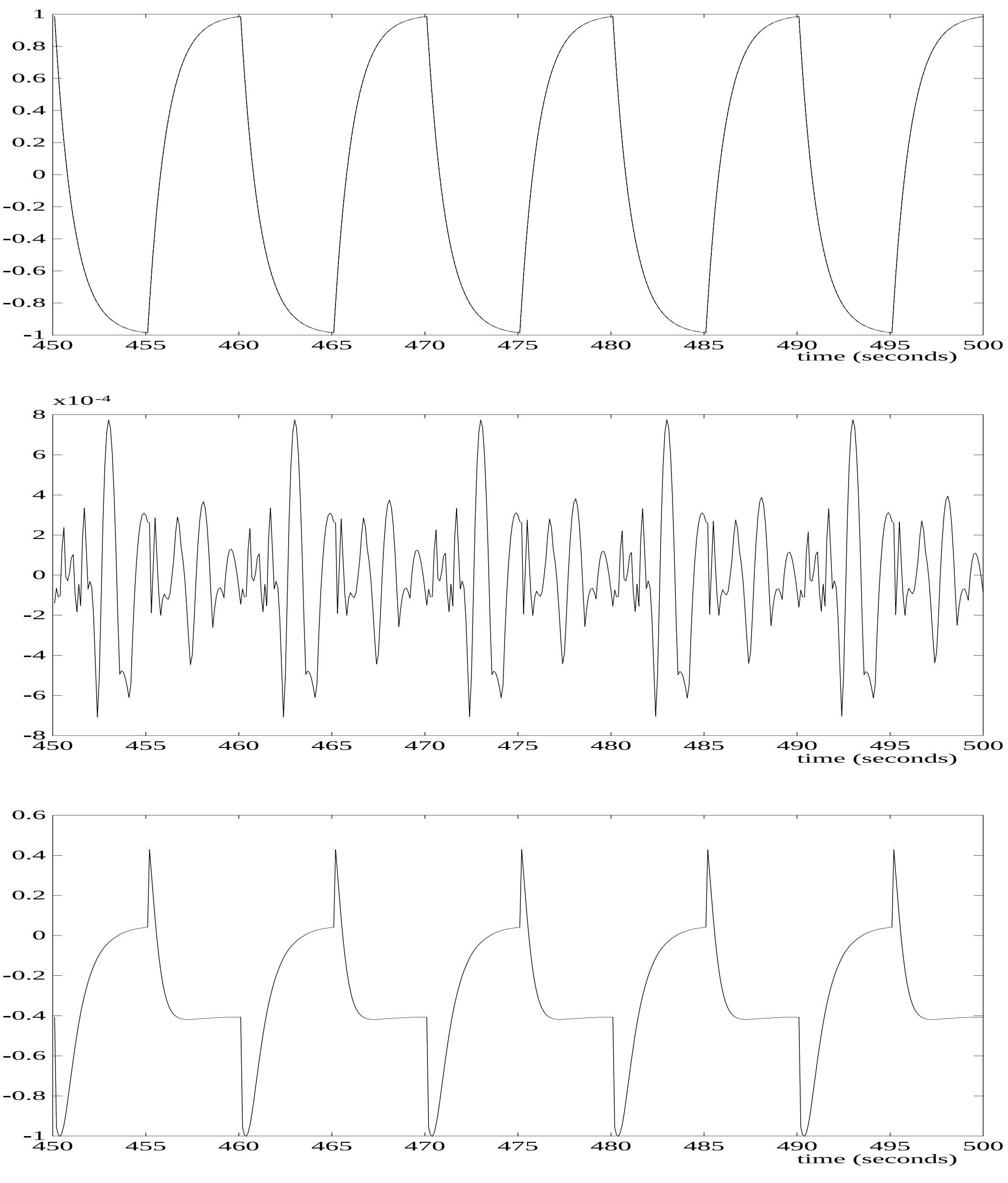

Figure 2: Tracking with a square wave reference after an identification phase. Top: the plant output (solid line) is indistinguishable from the model reference output (dashed line). Middle: the tracking error. Bottom: the control signal.

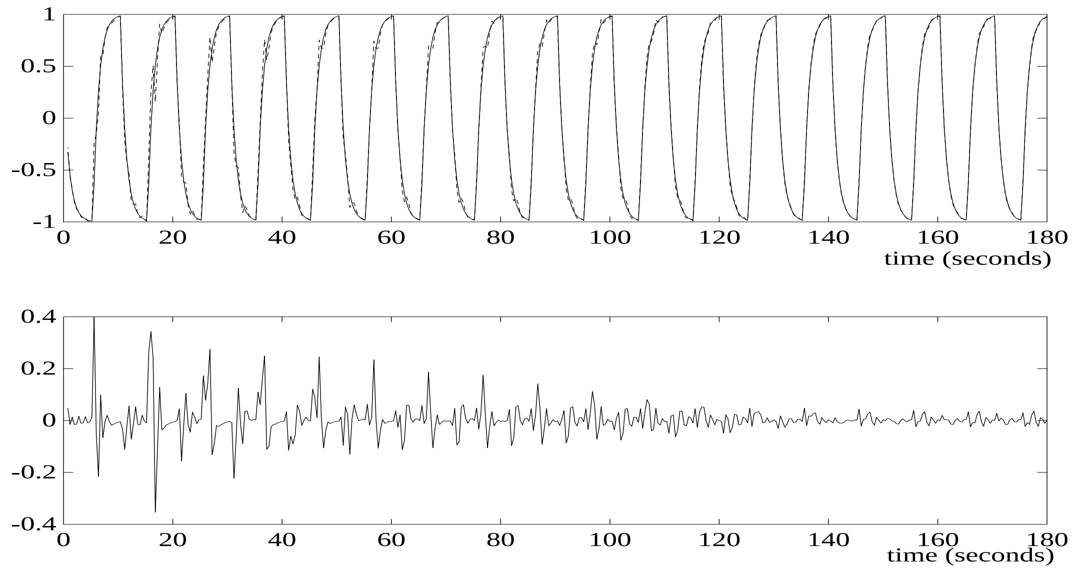

Figure 3: Tracking during first 3 minutes of operation with a square wave reference and no initial identification. Top: plant output (solid line) versus model reference output (dashed line). Bottom: tracking error.

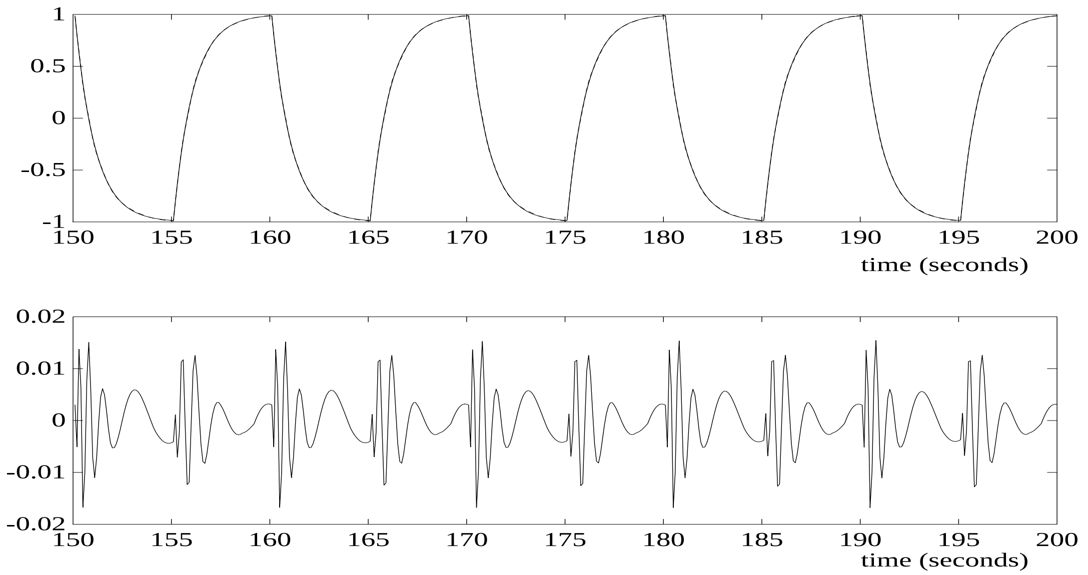

Figure 4: Tracking with a square wave reference after an initial control phase with this reference and without any identification phase. Top: plant output (solid line) versus model reference output (dashed line). Bottom: tracking error.

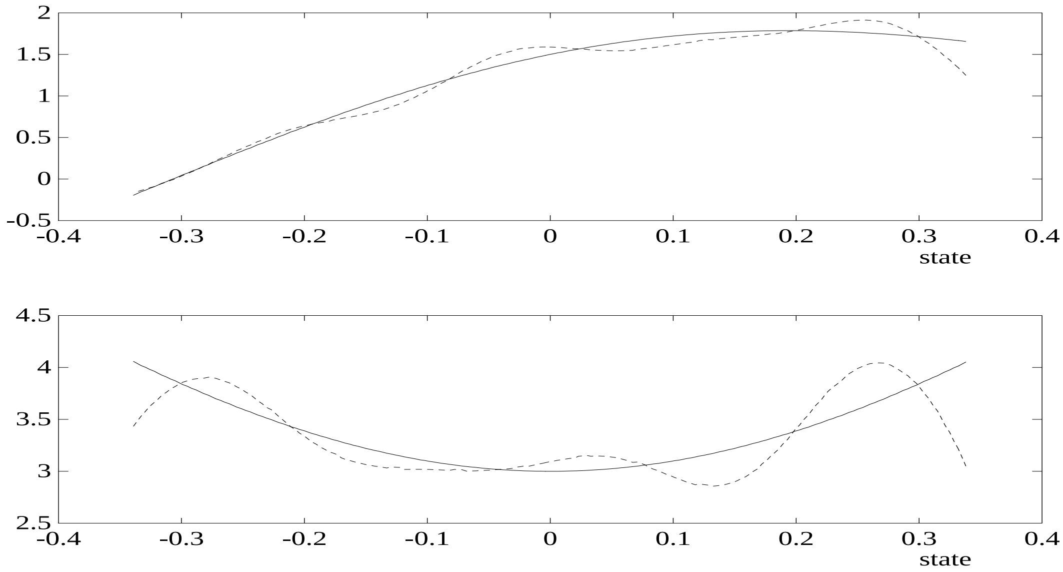

Figure 5: $L_f h(x)$ (top) and $L_g h(x)$ (bottom) approximations after 33 minutes of control with a square wave reference. The solid lines are the derivatives and the dashed lines their approximations.

4.2 Simulation 2

In this simulation we consider the control of the second order system given by

$$\begin{aligned} \dot{x}_1 &= \tfrac{x_2}{1+x_1^2} + 0.5x_2 \\ \dot{x}_2 &= 3\sin(3x_1x_2) + u \\ y &= x_1 \end{aligned} \tag{28}$$

It can be verified that $L_g h(x) = 0$ and $L_g L_f h(x)$ is positive and bounded away from zero for all x. Thus the plant has a strong relative degree $\gamma = 2$. The plant is minimum phase since there are no zero dynamics.

The two neural networks that we employ will aim to approximate the second order Lie derivatives $L_f^2 h(x)$ and $L_g L_f h(x)$. We use CMAC networks (Cerebellar Model Articulation Controller) [1] with gaussian receptive fields instead of square shaped fields normally used with the CMAC architecture. In each network there are two quantisation vectors for each state dimension; each quantisation vector contains 41 gaussian quantisation functions with standard deviation of 0.06 spread uniformly on the interval $[-2, 2]$, i.e. with a gap of 0.05 between adjacent gaussians. Thus there are altogether 3362 gaussians and the same number of weights in each network. Note that not all the receptive fields are necessarily activated during operation and that at any one time only a small number of receptive fields and weights participate in the computations, e.g. with these particular networks typically 24 gaussians are active at a time. Computation is therefore very fast and can be made even faster through the use of parallel computation.

We use the control law (11) with $v = \ddot{y}_m + \alpha_2(\dot{y}_m - \dot{y}) + \alpha_1(y_m - y)$ where $\alpha_2 = 3$ and $\alpha_1 = 2$. We filter the output error $e = y - y_m$ with the filter $\Psi(s) = s + 1$, so that the transfer function $\Omega(s) = \frac{s+1}{s^2+3s+2}$ is strictly positive real. Converting $\Omega(s)$ to state space representation we can compute the matrices P and Q. Doing this we find $\lambda_{\min} Q = 4$ which will be used to compute the error bound given in equation (24) shortly.

Our control objective is to track the output of a second order reference model given by

$$\begin{aligned} \dot{x}_m &= A_m x_m + b_m r \\ y_m &= c_m^T x_m \end{aligned} \tag{29}$$

where

$$A_m = \begin{bmatrix} -1.41 & -1 \\ 1 & 0 \end{bmatrix} \; b_m = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \; c_m = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

A hundred minutes long control/identification phase with a band limited noise reference and a learning rate of $\eta = 5$ was first applied. Figure 6 depict the resulting approximations of the Lie derivatives $L_f^2 h(x)$ (left) and $L_g L_f(x)$ (right). The top part of the figure corresponds to the actual derivatives and the bottom part to the approximations.

After the control/identification phase, parameters were held fixed and tracking with a square wave reference signal was attempted. The results are depicted in figure 7. The figure depicts, from top to bottom, the plant output versus the model reference output, the tracking error, the control signal, the $L_f^2 h(x)$ approximation error along the system trajectory and the $L_g L_f(x)$ approximation error along the system trajectory. It can be seen that the tracking error is in the range ± 0.04 and that the Lie derivatives approximation errors are quite large.

By offline approximation we found that in the relevant state space range, the networks approximation errors are $\bar{f} = 0.007$ and $\bar{g} = 0.009$. With a control bound $\bar{u} = 3$ we get $\bar{d} = 0.034$ (see equation (25)). Inserting this into (24) using the previously calculated eigenvalues we get $\bar{\epsilon} = 0.024$, and since $c^T = [1\ 1]$ (see equation 20), the corresponding tracking error bound is 0.034. The tracking error is approximately within this bound. In fact we can expect that the error may be further reduced since the derivative approximations are not accurate. Indeed, if we repeat the previous experiment but this time with parameter adaptation, the tracking error is reduced ten fold. Figure 8 depicts the results in this case. From top to bottom the figure shows the plant output versus the model reference output, the tracking error, and the Lie derivative approximations along the state trajectory. It can be seen that within a few seconds the tracking error and the approximation errors decrease considerably. This corresponds to further network training in the state regions activated by the specific trajectory. The approximation in other regions is not affected.

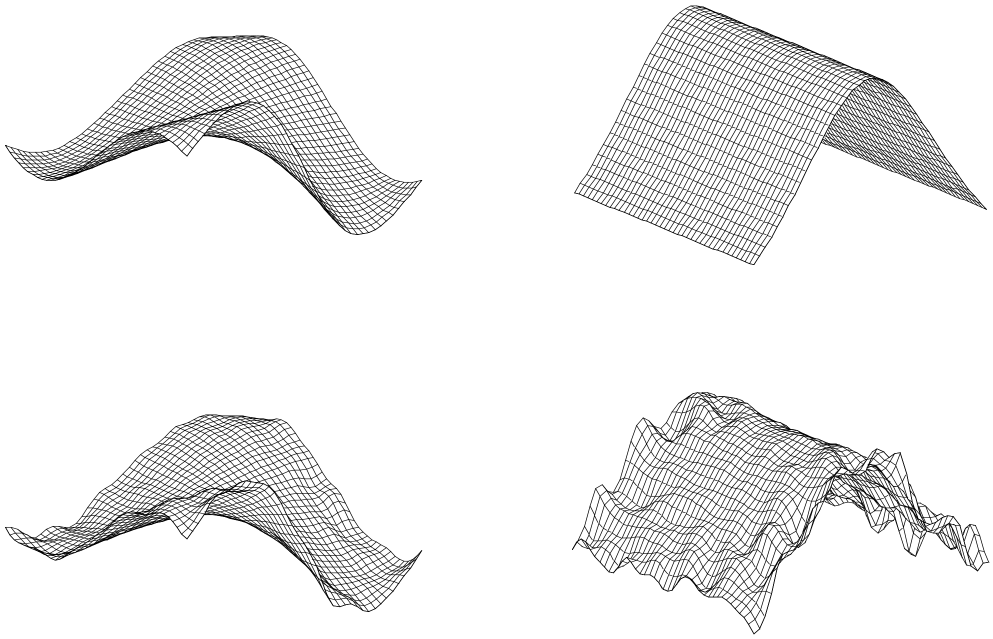

Figure 6: Lie derivative approximations after the control/identification phase. The left part corresponds to $L_f^2 h(x)$ and the right part to the $L_g L_f(x)$ derivative. The top depicts the derivatives and the bottom their approximations.

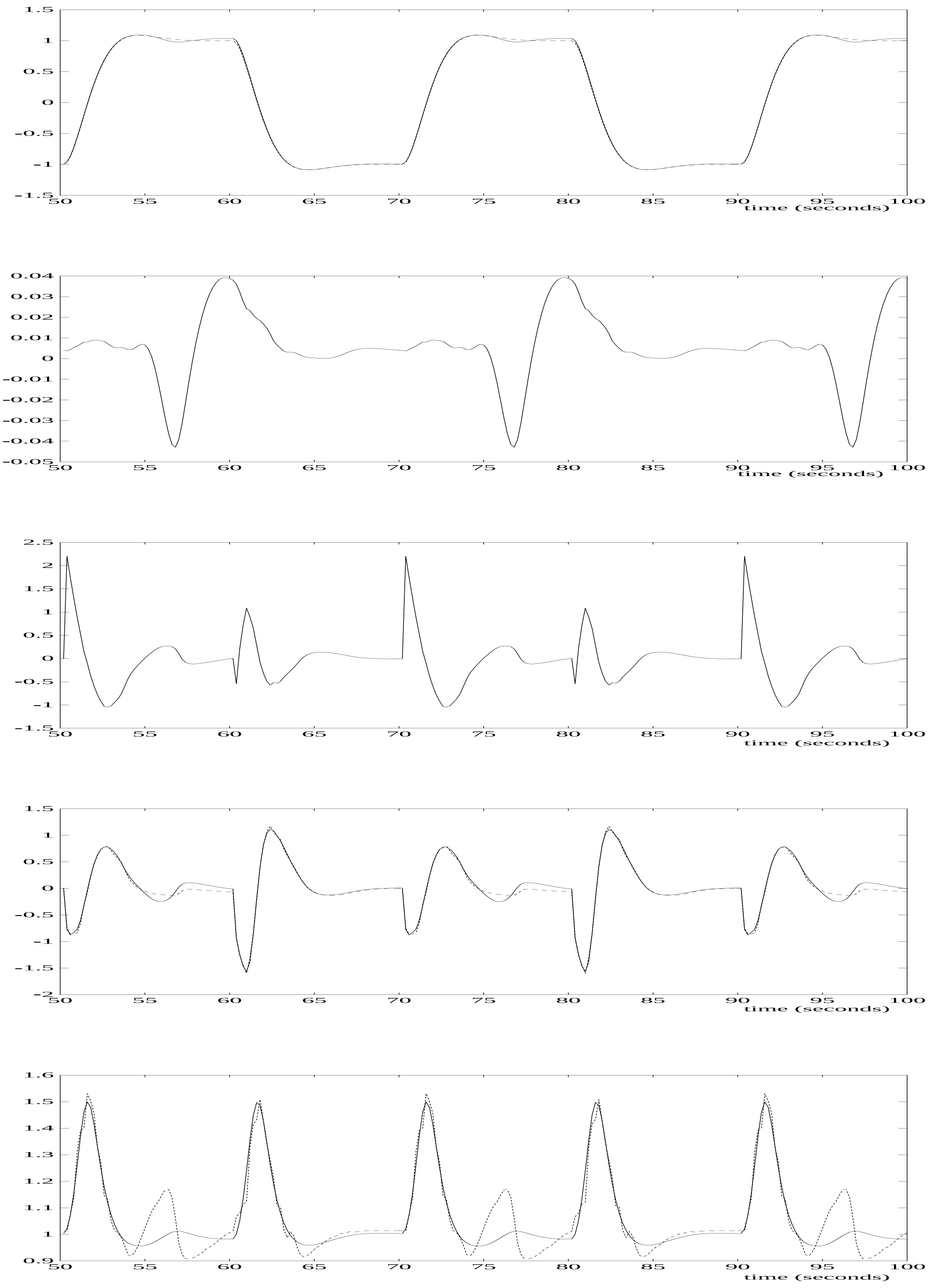

Figure 7: Tracking with a square wave reference, after identification phase, with adaptation switched off. Depicted from top to bottom are the plant output (solid line) versus the reference model output (dashed line), the tracking error, the control and the $L_f^2 h(x)$ and $L_g L_f(x)$ approximations along the systems' trajectory.

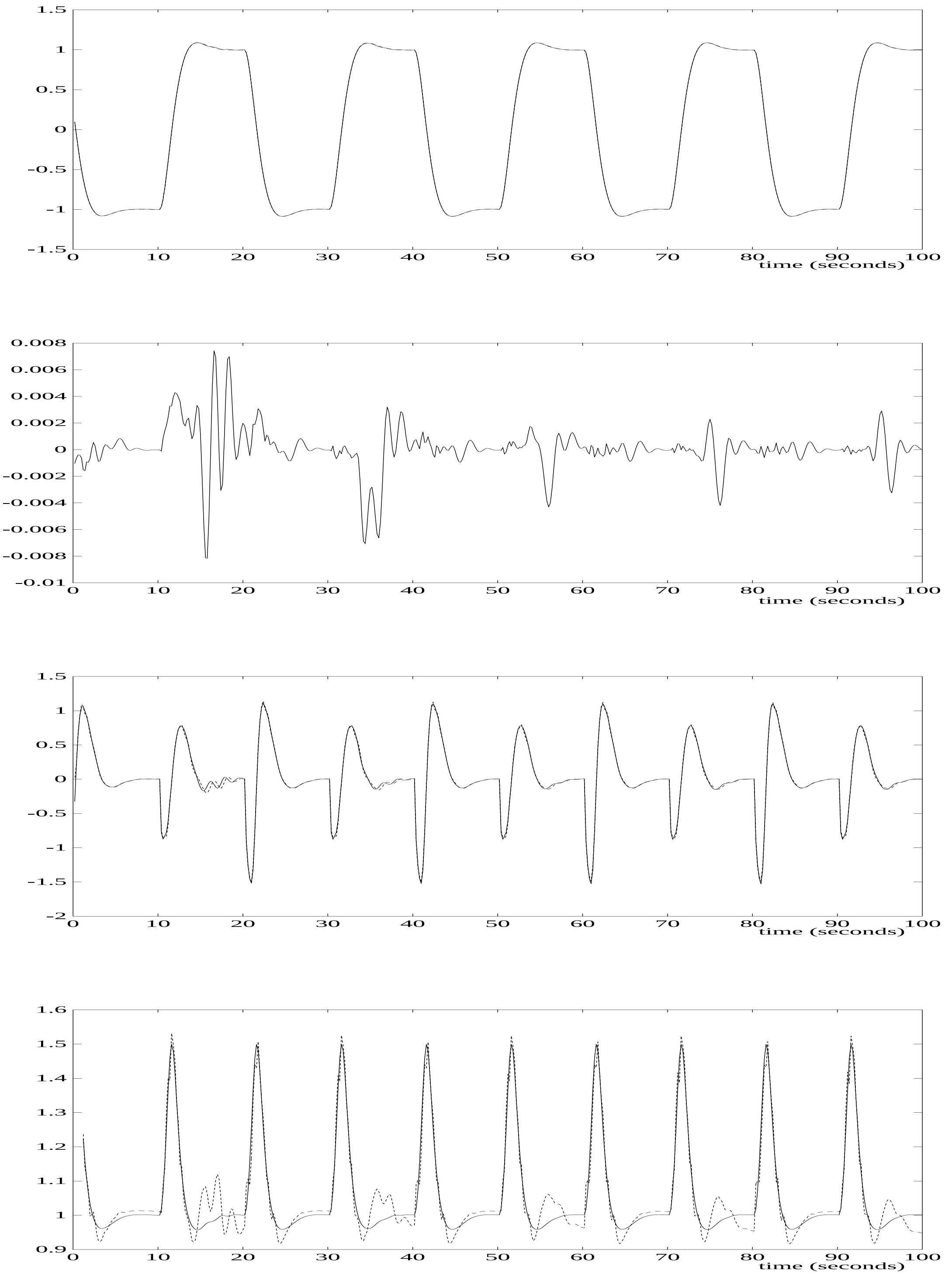

Figure 8: Performance with square wave reference, after identification phase, with adaptation switched on. Depicted from top to bottom are the plant output (solid line) versus the reference model output (dashed line), the tracking error, and the $L_f^2 h(x)$ and $L_g L_f(x)$ approximations along the systems' trajectory.

5 Extension to MIMO systems

In this section we shall see that under certain assumptions, the neural network control method introduced in the previous sections, can be extended in a straightforward manner to MIMO systems. We first, briefly introduce the generalization of some properties of SISO systems, to MIMO systems. The reader is referred to [6] for a detailed treatment.

Consider the system given by equation (1). The concept of a strong relative degree of this system is central in what follows. In order to introduce this concept we first define the $l \times l$ matrix

$$A(x) = \begin{bmatrix} L_{g_1} L_f^{\gamma_1 - 1} h_1(x) & \cdots & L_{g_l} L_f^{\gamma_1 - 1} h_1(x) \\ L_{g_1} L_f^{\gamma_2 - 1} h_2(x) & \cdots & L_{g_l} L_f^{\gamma_2 - 1} h_2(x) \\ \cdots & \cdots & \cdots \\ L_{g_1} L_f^{\gamma_l - 1} h_l(x) & \cdots & L_{g_l} L_f^{\gamma_l - 1} h_l(x) \end{bmatrix} \tag{30}$$

and the vector

$$B(x) = [L_f^{\gamma_1} h_1(x) \ldots L_f^{\gamma_l} h_l(x)]^T \tag{31}$$

where $R = [\gamma_1 \ldots \gamma_l]^T$ is a vector of positive integers.

The following definition is an extension of definition (1) to MIMO systems

Definition 2 *The system (1) is said to have* strong (vector) relative degree R, *if* $L_{g_j} L_f^k h_i(x) = 0$ *for all* $1 \leq i, j \leq l$, *for all* $1 \leq k < \gamma_i - 1$ *and for all* x, *and the matrix* $A(x)$ *is bounded away from singularity for all* x.

It is a fact that if a system has a strong relative degree R then $\gamma \triangleq \sum_{i=1}^{l} \gamma^i \leq n$.

Given the assumption that system (1) has a strong relative degree R, it is possible to show, by successive time differentiation of the outputs, that

$$y^{(R)} = B(x) + A(x)u \tag{32}$$

where $y^{(R)}$ denotes the vector of output derivatives $y^{(R)} = [y^{\gamma_1} \ldots y^{\gamma_l}]^T$ and $u = [u_1 \ldots u_l]^T$ is the input vector. This input/output equation is the extension of equation (5) to the MIMO case.

The linearizing static state feedback law equivalent to (6) is given by

$$u = A(x)^{-1}(v - B(x)) \tag{33}$$

where $v \in \Re^l$ is a secondary control vector. Note that definition (2) guarantees the invertibility of the matrix $A(x)$. This control law, when applied to the system (1), results in the linear input/output equation

$$y^{(R)} = v \tag{34}$$

This equation, being both linear and decoupled has the advantage of enabling one to use standard SISO control techniques for each of the decoupled subsystems separately.

A *state diffeomorphism* that transforms the system to a *normal form* can be introduced for MIMO systems, when the system has a strong relative degree. Similarly, the concept of *zero dynamics* can be defined in this case. These dynamics must be asymptotically stable if the system is to be controlled in stable manner.

The neural network control method is based on applying the state feedback control (33) with $A(x)$ and $B(x)$ replaced by their neural network approximations. The approximating networks are of the same type as the networks introduced in the SISO case.

We now present the assumptions, equivalent to assumptions 1–3, that are required for the extension of the neural network control method to MIMO systems

Assumption 4 *The plant has a known strong (vector) relative degree R, the zero dynamics are globally exponentially stable, the sign of* $\det[A(x)]$ *is known and a lower bound on* $|\det A(x)|$ *is known.*

Assumption 5 *The Lie derivatives that constitute the matrix $A(x)$ and the vector $B(x)$ are sufficiently regular that they can be uniformly approximated by neural networks of the form described by the product $\theta^T w(x)$, where $w(x)$ is a layer of receptive fields defined over the state space and θ is an adjustable parameter vector.*

Assumption 6 *The state x of the plant and the output derivatives up to order $R-1_l$ are available for measurement, where 1_l stands for the vector $[1\dots1]^T$ of length l.*

A convenient notational scheme for the control method is now introduced. Let $A_i(x)$ represent the i'th column of the matrix $A(x)$. Then equation (32) can be rewritten as

$$y^{(R)} = B(x) + \sum_{i=1}^{l} A_i(x)u_i \tag{35}$$

According to assumption (5) one can assume

$$\begin{array}{rcll} B(x) & = & \breve{\theta}_B w_B(x) + D_B(x) & \\ A_i(x) & = & \breve{\theta}_{A_i} w_{A_i}(x) + D_{A_i}(x) & 1 \le i \le l \\ \text{with} & & & \\ \|D_B(x)\| & \le & \bar{B} \text{ for all } x \in X & \\ \|D_{A_i}(x)\| & \le & \bar{A}_i \text{ for all } x \in X & 1 \le i \le l \end{array} \tag{36}$$

where $\breve{\theta}_B$, $\breve{\theta}_{A_i} \in \Re^{l\times p}$ are *optimal* parameter matrices, $w_B(x)$, $w_{A_i}(x) \in \Re^p$ are vectors of basis functions, $D_B(x)$, $D_{A_i}(x) : \Re^n \to \Re^l$ are disturbance vectors and $\bar{B}$ and $\bar{A}_i$ are approximation bounds defined in a similar sense to the corresponding bounds in the SISO case [7], i.e equations (9) and (10).

The neural networks that approximate $B(x)$ and $A(x)$ are given by

$$\begin{array}{rcll} \hat{B}(x) & = & \theta_B w_B(x) & \\ \hat{A}_i(x) & = & \theta_{A_i} w_{A_i}(x) & 1 \le i \le l \end{array} \tag{37}$$

[7] For notational convenience we have assumed that each vector of Lie derivatives is approximated by networks with identical basis functions

where θ_B, $\theta_{A_i} \in \Re^{l \times p}$ are parameter matrices.

The control law implemented with these networks is

$$u = \hat{A}(x)^{-1}(v - \hat{B}(x)) \tag{38}$$

The approximation matrix $\hat{A}(x)$ must be kept bounded away from singularity at all times. This can be achieved by adding an auxiliary term to the adaptive law, to push $\hat{A}(x)$ away from singularity, whenever $|\det A(x)|$ is small. This is where the lower bound on $|\det A(x)|$ in assumption (4) is needed.

We now define the overall optimal parameter matrix $\breve{\Theta} = [\breve{\theta}_B \ \breve{\theta}_{A_1} \ldots \breve{\theta}_{A_l}]$, the adjustable parameter matrix $\Theta = [\theta_B \ \theta_{A_1} \ldots \theta_{A_l}]$, the parameter error matrix $\Phi = \Theta - \breve{\Theta}$, the regressor vector $W^T = [-w_B^T \ -w_{A_1}^T u_1 \ldots -w_{A_l}^T u_l]$ and the disturbance vector $D(x,u) = D_B(x) + \sum_{i=1}^{l} D_{A_i} u_i$. Inserting the control (38) into equation (35) and using these definitions, one can obtain the equation

$$y^{(R)} = \Phi W(x,u) + D(x,u) + v \tag{39}$$

which is equivalent to equation (13) in the SISO case.

With the objective of tracking the reference signal $y_m \in \Re^l$, the control v is defined for each element individually in exactly the same way as was done in equation (14)

$$v_i = y_{mi}^{(\gamma_i)} + \alpha_{i\gamma_i}(y_{mi}^{(\gamma_i-1)} - y_i^{(\gamma_i-1)}) + \cdots + \alpha_{i1}(y_{mi} - y_i) \ \ 1 \le i \le l \tag{40}$$

Inserting v into equation (39) and using the error definition $e = [e_1 \ldots e_l]^T$ where $e_i = y_i^{(\gamma_i)} - y_{mi}^{(\gamma_i)}$ one gets the error equation

$$e = \Gamma^{-1}\{\Phi W(x,u) + D(x,u)\} \tag{41}$$

where $\Gamma^{-1}(s) = diag(\Gamma_1^{-1}(s), \ldots, \Gamma_l^{-1}(s))$ with $\Gamma_i(s) = s^{(\gamma_i)} + \alpha_{i\gamma_i}s^{(\gamma_i-1)} + \cdots + \alpha_{i1}$.

In order to derive a stable parameter adaptation law, the auxiliary error $e_1 \in \Re^l$ is first defined by $e_1 = \Psi(s)e$ where $\Psi(s) = diag(\psi_1(s), \ldots, \psi_l(s))$ and $\psi_i(s) = b_{i\gamma_i}s^{(\gamma_i-1)} + \cdots + b_{i1}$. $\Psi(s)$ is defined in such a way that the transfer functions $\psi_i(s)\Gamma_i^{-1}(s)$ are SPR for all $1 \le i \le l$. Therefore, the equation for the auxiliary error is

$$e_1 = \Omega(s)\{\Phi W(x,u) + D(x,u)\} \tag{42}$$

where $\Omega(s) = \Psi(s)\Gamma^{-1}(s)$ is a strictly positive real transfer matrix [8].

It is now possible to define the parameter update law

$$\dot{\Theta} = -\Sigma^{-1} e_1 W(x,u)^T \tag{43}$$

where $\Sigma \in \Re^{l \times l}$ is a constant symmetric positive definite matrix that corresponds to the learning rate in the SISO case.

The SPR error equation (42) can be represented in state equations of the form

$$\begin{aligned} \dot{\epsilon} &= A\epsilon + B\{\Phi^T W(x,u) + D(x,u)\} \\ e_1 &= C\epsilon \end{aligned} \tag{44}$$

[8] Using the fact that the diagonal elements of $\Omega(s)$ are SPR by construction, it is not difficult to show that the overall transfer matrix $\Omega(s)$ is SPR

where $\epsilon \in \Re^\gamma$, $A \in \Re^{\gamma\times\gamma}$, $B, C^T \in \Re^{\gamma\times l}$ and A, B and C satisfy

$$\begin{array}{rcl} PA + A^T P & = & -Q \\ PB & = & C^T \end{array} \tag{45}$$

where $P \in \Re^{\gamma\times\gamma}$ and $Q \in \Re^{\gamma\times\gamma}$ are symmetric positive definite matrices. With this representation one can define the Lyapunov function that corresponds to the error equation (42) and the parameter update law (43)

$$V = \epsilon^T P \epsilon + Tr(\Phi^T \Sigma \Phi) \tag{46}$$

where $Tr(\cdot)$ denotes the matrix trace operator.

By differentiating the Lyapunov function along the system's trajectory we get, after some manipulations

$$\dot{V} = -\epsilon^T Q \epsilon + 2\epsilon^T C^T D(x, u) \tag{47}$$

and from this equation one can concludes that $\dot{V}$ is negative as long as

$$\|\epsilon^T\| \geq 2\frac{\sqrt{\lambda_{\max}(CC^T)}}{\lambda_{\min} Q}\bar{D} \triangleq \bar{\epsilon} \tag{48}$$

where $\bar{D}$ is an upper bound of the inherent approximation error $D(x, u)$. This tracking error bound is equivalent to the bound given in equation (25) for the SISO algorithm. It has the same significance because it quantifies the connection between the network approximation error bounds and the tracking error in a similar way.

Analysis of the persistent excitation of the vector $W^T = [-w_B^T \ - w_{A_1}^T u_1 \ldots - w_{A_l}^T u_l]$ on the algorithm behaviour is also similar to the SISO case. Here however, the regressor vector is larger and substantially more complicated. This means that it is more difficult to achieve persistent excitation in the MIMO case. However, the main conclusions from the PE analysis in the SISO case carry over to MIMO systems, e.g. the need for an initial control/identification phase, the significance of the reference signal richness, and the parameters behaviour with a local receptive field representation when the regressor is not sufficiently excited. The initial identification/control phase, in the MIMO case, may need to be extended for a long period of time in order to achieve a good initial parameter convergence.

In addition to the increased complexity in the MIMO case, another difficulty that arises is parasitic coupling between subsystems. The inherent approximation errors imply that even when the parameters converge to their optimal values, there is still some parasitic coupling between the subsystems. The approximation ability of the networks becomes important in this sense too.

In many applications, the relevant state space constitutes only a small subset in $\Re^n$ and therefore the use of efficient ways to reduce the number of parameters, e.g. hashing schemes, becomes highly desirable.

6 Conclusions and Future Work

A direct control scheme for the nonlinear system (1) using neural networks was presented. The method is based on neural network approximation of certain Lie derivatives of the nonlinear functions that define the system. Adaptation of the networks takes place during control and is based on a stability type method. The control method's main advantage over other techniques is that it does not assume knowledge of the plant nonlinearities. Only some general properties of the plant are assumed to be known.

The network learning rule was analysed in light of its close relation to adaptive control methods. This analysis provided an interesting bridge between well studied, rigorous adaptive control methods and the field of neural network training. In particular, the importance of the property of persistent excitation and its implications to learning with networks of localized receptive fields was discussed.

An important conclusion from the persistent excitation analysis is the desirability of inclusion of an initial control/identification phase. This initial phase allows for a good initial approximation of the Lie derivatives which in turn guarantee good initial tracking with any specialized reference input. The inclusion of an initial control/identification phase may require in some cases the use of a specifically designed reference signal and may take considerable time. We have also seen that adaptation can proceed with a non-rich reference input, with the result of improved approximation in some state space regions with other regions remaining unaffected. This phenomenon was related to the localized nature of the network.

Another conclusion of the analysis is the strong dependency of the method on the approximation abilities of the networks. This dependency was quantified through the introduction of the error bound presented in equations (24) and (48). In some cases, the Lie derivatives may be too irregular and complex so that good approximation by the type of networks described may be impractical. Prior information about the system nonlinearities may be of great help in the networks design.

The CMAC neural network with gaussian receptive fields was proposed, in simulation 4.2, as a candidate for the approximating networks. The use of a CMAC network allows fast processing with a relatively small amount of memory, especially if implemented on a parallel machine. Also, hashing schemes can be used to reduce the amount of memory involved [7].

The control method proposed is not directly applicable to the discrete time case in which the theory of exact linearization is far more complicated [8]. More research is needed to develop neural network control methods for this case. Chen [3] has reported a control scheme for the discrete time case with a restricted type of plant.

Further research is needed on the convergence, stability and other issues related to the proposed control method. It is an interesting question if ways of adapting the basis functions themselves, i.e location, width and shape of the receptive fields, can be developed, or if more global receptive fields can be effectively incorporated into the networks. Another related question is whether one can employ multilayer perceptrons for the approximation. These and related issues will be the subject of further research.

7 Acknowledgements

I wish to thank my supervisor Prof. Frank Fallside for his support and to Visakan Kadirkamanathan, David Bijl, Philip Steen, Tim Jervis and Ian Braithwaite for their helpful comments.

References

[1] J. A. Albus. *Brains, Behaviour and Robotics.* BYTE Books, 1981.

[2] B. D. O. Anderson and S. Vongpanitlerd. *Network Analysis and Synthesis.* Prentice-Hall, Englewood Cliffs, New Jersey, 1973.

[3] F. C. Chen. Backpropagation neural networks for nonlinear self-tuning adaptive control. *IEEE Control Systems Magazine*, 3:44–47, 1990.

[4] D. Claude. Everything you always wanted to know about linearization. In M. Fliess and M. Hazewinkel, editors, *Algebraic and Geometric Methods in Nonlinear Control Theory.* D. Reidel, 1986.

[5] R. A. Horn and C. R. Johnson. *Matrix analysis.* Cambridge University Press, Cambridge, England, 1990.

[6] A. Isidori. *Nonlinear Control Systems: An Introduction.* Springer-Verlag, Berlin, 1989.

[7] W. T. Miller, F. H. Glanz, and L. G. Kraft. CMAC: an associative neural network alternative to backpropagation. *Proceeding of the IEEE*, 78:1561–1567, 1990.

[8] S. Monaco and D. Normand-Cyrot. Nonlinear systems in discrete time. In M. Fliess and M. Hazewinkel, editors, *Algebraic and Geometric Methods in Nonlinear Control Theory.* D. Reidel, 1986.

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

[10] M. Niranjan and F. Fallside. Neural networks and radial basis functions in classifying static speech patterns. Technical Report CUED/F-INFENG/TR 22, Cambridge University Engineering Department, Cambridge, England, 1988.

[11] M. J. D. Powell. Radial basis functions for multi-variable interpolation: a review. Technical Report DAMTP/NA12, Department of Applied Mathematics and Theoretical Physics, Cambridge University, Cambridge, England, 1985.

[12] S. Sastry and M. Bodson. *Adaptive Control: Stability Convergence and Robustness.* Prentice-Hall International, Englewood Cliffs, New Jersey, 1989.

[13] A. Teel, R. Kadiyala, P. Kokotovik, and S. Sastry. Indirect techniques for adaptive input-output linearization of non-linear systems. *Int. J. Control*, 53:193–222, 1991.