Chapter 8

Neural Network Control of Robots

A framework is given for controller design using neural networks. These structures possess a universal approximation property that allows them to be used in feedback control of unknown systems without requirements for linearity in the system parameters or finding a regression matrix. Feedback control topologies and weight tuning algorithms are given here that guarantee closed-loop stability and bounded weights.

8.1 Introduction

In recent years, there has been a great deal of effort to design feedback control systems that mimic the functions of living biological systems [White and Sofge 1992], [Miller et al. 1991]. There has been great interest recently in 'universal model-free controllers' that do not need a mathematical model of the controlled plant, but mimic the functions of biological processes to learn about the systems they are controlling on-line, so that performance improves automatically. Techniques include fuzzy logic control, which mimics linguistic and reasoning functions, and artificial neural networks, which are based on biological neuronal structures of interconnected nodes. Neural networks (NNs) have achieved great success in classification, pattern recognition, and other open-loop applications in digital signal processing and elsewhere. Rigorous analyses have shown how to select NN topologies and weights, for instance, to discriminate between specified exemplar patterns. By now, the theory and applications of NN in open-loop applications are well understood, so that

NN have become an important tool in the repertoire of the signal processor and computer scientist.

There is a large literature on NN for feedback control of unknown plants. Until the 1990's, design and analysis techniques were ad hoc, with no repeatable design algorithms or proofs of stability and guaranteed performance. In spite of this, simulation results appearing in the literature showed good performance. Most of the early approaches used standard backpropagation weight tuning [Werbos 1992] since rigorous derivations of tuning algorithms suitable for closed-loop control purposes were not available. Many NN design techniques mimicked adaptive control approaches, where rigorous analysis results were available [Åström and Wittenmark 1989], [Landau 1979], [Goodwin et al. 1984], proposing NN feedback control topologies such as indirect identification-based control, inverse dynamics control, series-parallel techniques, etc.

In keeping with the philosophy of those working in control system theory since Maxwell, Lyapunov, A.N.Whitehead, von Bertalanffy, and others, to address such issues it is necessary to begin with the knowledge available about the system being controlled. Narendra [Narendra 1991], [Narendra and Parthasarathy 1991] and others [Miller et al. 1991], [White and Sofge 1992], [Werbos 1992] pioneered rigorous NN controls applications by studying the dynamical behavior of NN in closed-loop systems, including computation of the gradients needed for backprop tuning. Other groups providing rigorous analysis and design techniques for closed-loop NN controllers included [Sanner and Slotine 1991] who showed how to use radial basis function NN in feedback control, [Chen and Khalil 1992], [Chen and Liu 1994] who provided NN tuning algorithms based on dead-zone methods, [Polycarpou and Ioannou 1991], [Polycarpou 1996] who used projection methods, [Lewis et al. 1993], [Lewis et al. 1995] who used an e-mod approach, [Sadegh 1993] who provided NN controllers for discrete-time systems, and [Rovithakis and Christodoulou 1994] who used dynamic NN for feedback control.

In this chapter is given an approach to the design and analysis of *neural network* controllers based on several PhD dissertations and a body of published work, including [Lewis et al. 1999], [Kim and Lewis 1998]. The control structures discussed here are *multiloop controllers* with NN in some of the loops and an outer tracking unity-gain PD loop. There are repeatable design algorithms and guarantees of system performance including both small tracking errors and bounded NN weights. It is shown that as uncertainty about the controlled system increases or performance requirements increase, the NN controllers require more and more structure. An exposition of this material in greater depth appears in [Lewis 1999], which also shows relationships with Fuzzy Logic Systems.

8.2 Background in Neural Networks

In this section is provided the background in neural network structures required for feedback control. For more details see [Haykin 1994], [Lewis et al. 1999], [Lewis and Kim 1998]. Two key features that make NN useful for feedback control are their *universal approximation property* and their *learning capability*, which arises due to the fact that their weights are tunable parameters that can be updated to improve controller performance. The universal approximation property is the main feature that makes non-linear network structures more suitable for robot control than adaptive robot controllers, which have generally relied upon the determination of a regression matrix, which in turn requires linearity in the tunable parameters (LIP).

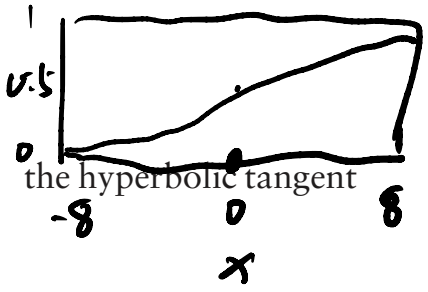
Multilayer Neural Networks

A multilayer neural network is shown in Figure 8.2.1. This NN has two layers of adjustable weights, and is called here a 'two-layer' net. This NN has no internal feedback connections and so is termed *feedforward*, and no internal dynamics and so is said to be *static*. The NN output y is a vector with m components that are determined in terms of the n components of the input vector x by the recall equation

$$y_i = \sum_{j=1}^{L} \left[w_{ij} \sigma \left(\sum_{k=1}^{n} v_{jk} x_k + \theta_{v_j} \right) + \theta_{w_i} \right]; \quad i = 1, \ldots, m \tag{8.2.1}$$

where $\sigma(\cdot)$ are the *activation functions* and L is the number of *hidden-layer neurons*. The first-layer interconnections weights are denoted v_{jk} and the second-layer interconnection weights by w_{ij}. The threshold offsets are denoted by θ_{vj}, θ_{wi}.

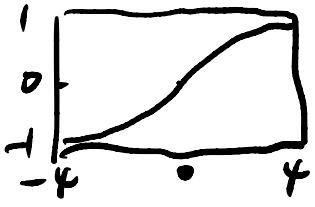
Many different activation functions $\sigma(\cdot)$ are in common use. For feed-back control using multilayer NN it is required that $\sigma(\cdot)$ be smooth enough so that at least its first derivative exists. Suitable choices include the sigmoid



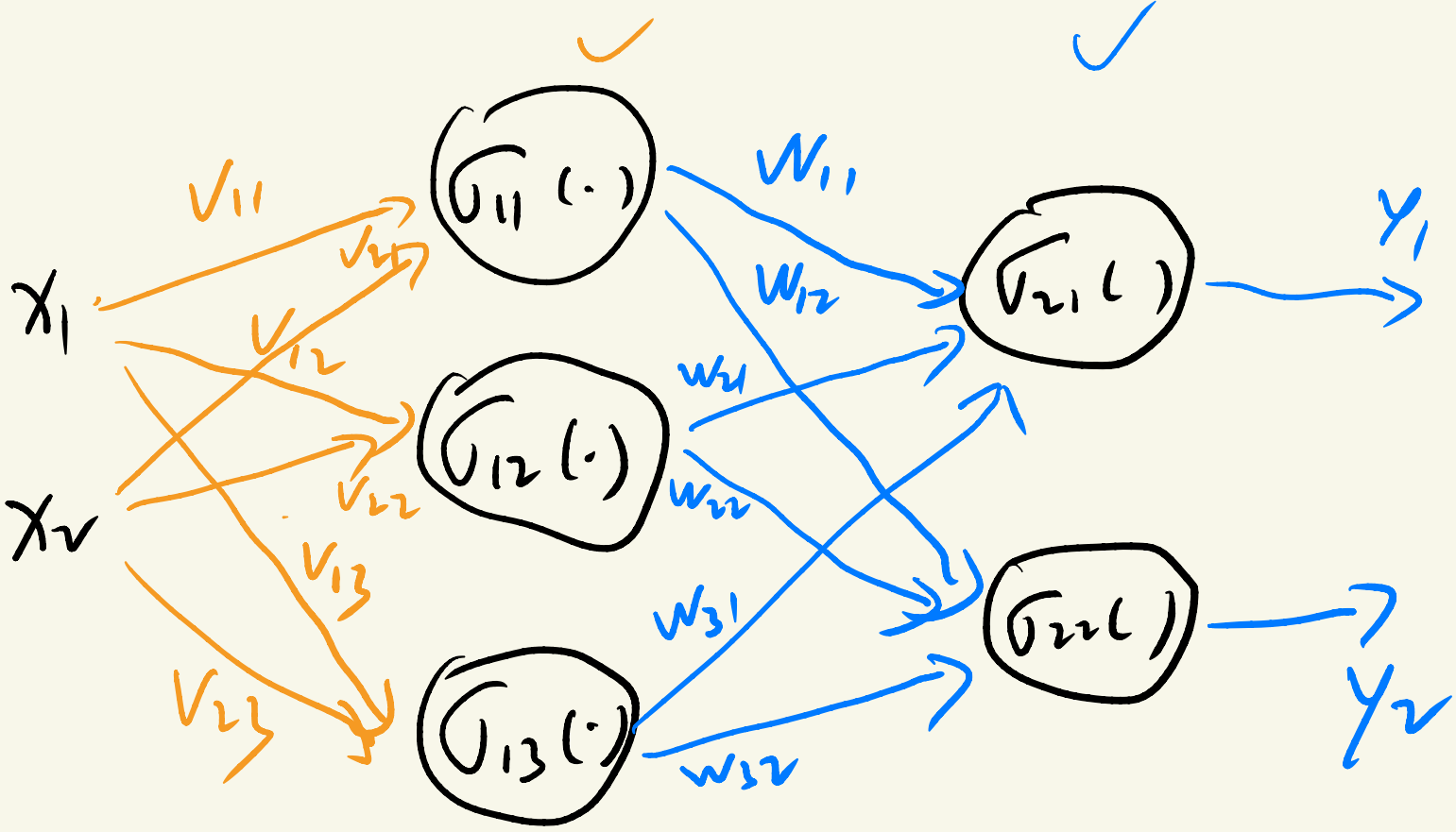
$$\sigma(x) = \frac{1}{1 + e^{-x}}, \tag{8.2.2}$$

the hyperbolic tangent

$$\sigma(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}, \tag{8.2.3}$$



and other logistic-curve-type functions, as well as the gaussian and an assortment of other functions.



$$y_1 = \sigma_{21}\left(w_{11}\sigma_{11}(\ ) + w_{21}\sigma_{12}(\ ) + w_{31}\sigma_{13}(\ ) \right)$$

$v_{11}x_1 + v_{12}x_2$

$v_{12}x_1 + v_{22}x_2$

$v_{13}x_1 + v_{23}x_2$

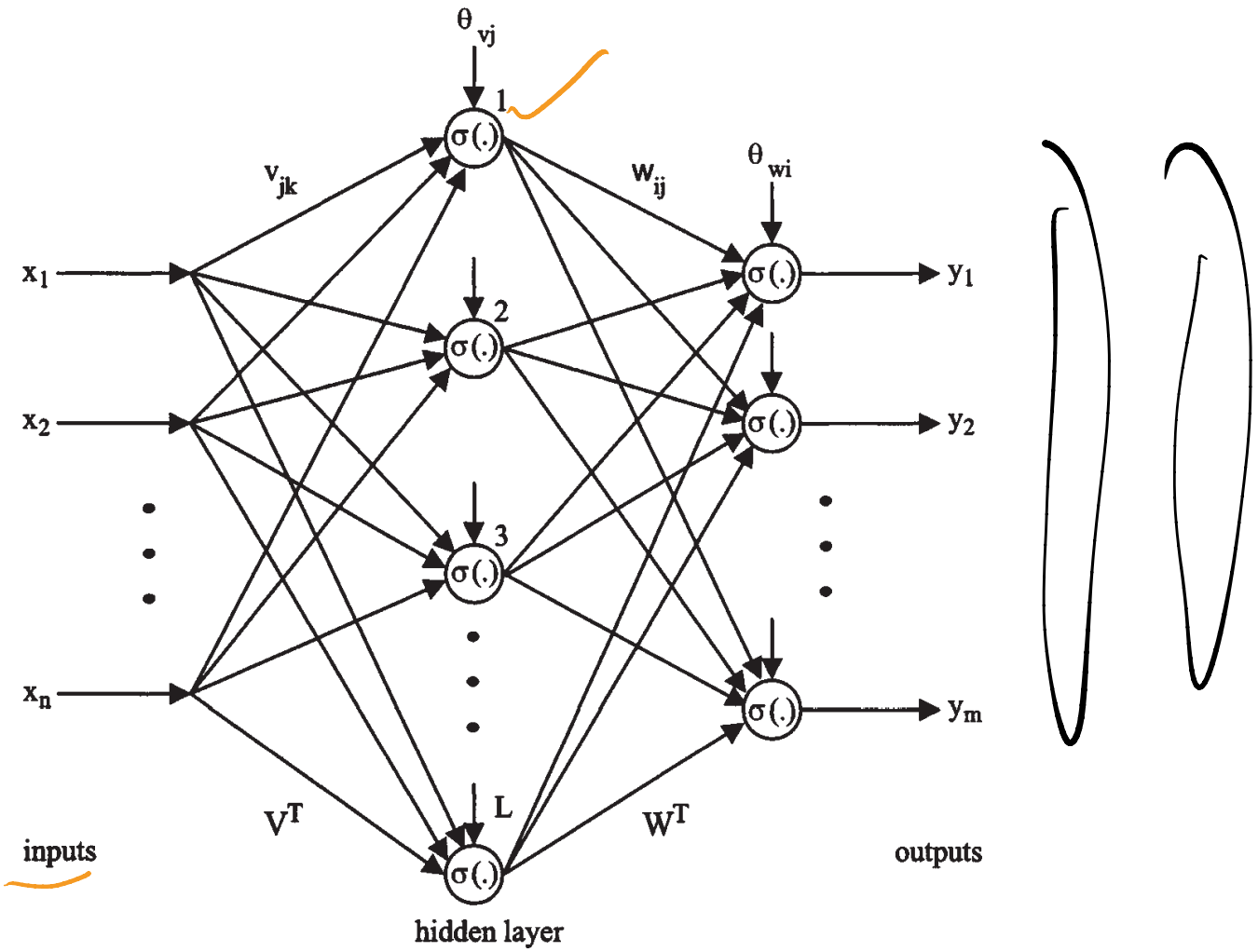


Figure 8.2.1: Two-layer feedforward neural network.

By collecting all the NN weights v_{jk}, w_{ij} into matrices of weights V^T, W^T, the NN recall equation may be written in terms of vectors as

$$y = W^T \sigma(V^T x). \tag{8.2.4}$$

The thresholds are included as the first columns of the weight matrices W^T, V^T; to accommodate this, the vectors x and $\sigma(\cdot)$ need to be augmented by placing a '1' as their first element (e.g. $x=[1\ x_1\ x_2 \ldots x_n]^T$). In this equation, to represent (8.2.1) one has sufficient generality if $\sigma(\cdot)$ is taken as a diagonal function from $\Re^L$ to $\Re^L$ that is $\sigma(z)=diag\{\sigma(z_j)\}$ for a vector $z=[z_1 z_2 \ldots z_L]^T \in \Re^L$.

Note that the recall equation is *nonlinear* in the first-layer weights and thresholds V.

Universal Function Approximation Property. NN satisfy many important properties. A main one of concern for feedback control purposes is the *universal function approximation property* [Hornik and Stinchombe 1989]. Let $f(x)$ be a general smooth function from $\Re^n$ to $\Re^m$. Then, it can be shown that, as long as x is restricted to a compact set S of $\Re^n$, there exist weights and thresholds such that one has

$$f(x) = W^T \sigma(V^T x) + \epsilon \tag{8.2.5}$$

for some number of hidden-layer neurons L. This holds for a large class of activation functions, including those just mentioned. The value ϵ is called the *NN functional approximation error,* and it generally decreases as the net size L increases. In fact, for any choice of a positive number ϵ_N, one can find a feedforward NN such that

$$\|\epsilon\| < \epsilon_N \tag{8.2.6}$$

for all x in S. The selection of the net size L for a prescribed accuracy ϵ_N is an open question for general unstructured fully-connected NN. This problem can be solved for CMAC, FL nets, and other structured nonlinear nets as subsequently described.

The ideal NN weights in matrices W, V that are needed to best approximate a given nonlinear function $f(x)$ are difficult to determine. In fact, they may not even be unique. However, all one needs to know for controls purposes is that, for a specified value of ϵ_N *some ideal approximating NN weights exist*. Then, an *estimate* of $f(x)$ can be given by

$$\hat{f}(x) = \hat{W}^T \sigma(\hat{V}^T x) \tag{8.2.7}$$

where W, V are estimates of the ideal NN weights that are provided by some on-line weight tuning algorithms subsequently to be detailed. Note that all 'hat' quantities are known, since they may be computed using measurable signals and current NN weight estimates.

The assumption that there exist ideal weights such that the approximation property holds is very much like various similar assumptions in adaptive control [Åström and Wittenmark 1989], [Landau 1979], including *Erzberger's assumptions* and *linearity in the parameters*. The very important difference is that in the NN case, *the universal approximation property always holds,* while in adaptive control such assumptions often do not hold in practice, and so they imply restrictions on the form of the systems that can be controlled.

Overcoming the NLIP Problem. Multilayer NN are nonlinear in the weights

V and so weight tuning algorithms that yield guaranteed stability and bounded weights in closed-loop feedback systems have been difficult to discover until a few years ago. The NLIP problem is easily overcome if a correct and rigorous approach is taken. One approach is the following appropriate use of the Taylor series [Lewis et al. 1996].

Define the functional estimation error

$$\tilde{f} = f - \hat{f}, \tag{8.2.8}$$

the weight estimation errors

$$\tilde{W} = W - \hat{W}, \quad \tilde{V} = V - \hat{V}, \tag{8.2.9}$$

and the hidden-layer output error

$$\tilde{\sigma} = \sigma - \hat{\sigma} \equiv \sigma(V^T x) - \sigma(\hat{V}^T x). \tag{8.2.10}$$

For any z one may write the Taylor series

$$\sigma(z) = \sigma(\hat{z}) + \sigma'(\hat{z})\tilde{z} + O(\tilde{z})^2, \tag{8.2.11}$$

where σ' is the jacobian and the last term means terms of order z^2.Therefore,

$$\tilde{\sigma} = \sigma'(\hat{V}^T x)\tilde{V}^T x + O(\tilde{V}^T x)^2 \equiv \hat{\sigma}'\tilde{V}^T x + O(\tilde{V}^T x)^2. \tag{8.2.12}$$

This key equation allows one to write the functional estimation error as

$$\begin{aligned}
\tilde{f} =& f - \hat{f} = W^T\sigma\left(V^T x\right) - \hat{W}^T\sigma\left(\hat{V}^T x\right) + \varepsilon \\
=& \tilde{W}^T\sigma(\hat{V}^T x) + W^T[\sigma(V^T x) - \sigma(\hat{V}^T x)] + \epsilon \\
=& \tilde{W}^T\hat{\sigma} + W^T[\hat{\sigma}'\tilde{V}^T x + O(\tilde{V}^T x)^2] + \epsilon \\
=& \tilde{W}^T[\hat{\sigma} - \hat{\sigma}'\hat{V}^T x] + \hat{W}^T\hat{\sigma}'\tilde{V}^T x + \tilde{W}^T\hat{\sigma}'V^T x \\
& + W^T O(\tilde{V}^T x)^2 + \epsilon.
\end{aligned} \tag{8.2.13}$$

The first term has W multiplied by a known quantity (in square braces), and the second term has V multipled by a known quantity. When used subsequently in the closed-loop error dynamics, this form allows one to derive tuning laws for V and W that guarantee closed-loop stability.

Linear-in-the-parameter neural nets

If the first-layer weights and thresholds V in (8.2.4) are fixed and only the second-layer weights and thresholds W are tuned, then the NN has only one layer of tunable weights. One may then define the fixed function $\phi(x) = \sigma(V^T x)$ so that such a 1-layer NN has the recall equation

$$y = W^T \phi(x), \tag{8.2.14}$$

where $x \in \Re^n, y \in \Re^m, \phi(\cdot) : \Re^n \to \Re^L$ and L is the number of hidden-layer neurons. This NN is linear in the tunable parameters W, so that it is far easier to tune. In fact, standard adaptive control proofs can be used to derive suitable tuning algorithms.

Though LIP, these NN still offer an enormous advantage over standard adaptive control approaches that require the determination of a regression matrix since they satisfy a universal approximation property if the functions $\phi(\cdot)$ are correctly chosen. Note that adaptive controllers are linear in the *system* parameters, while 1-layer NN are linear in the *NN weights*. An advantage of 1-layer NN over 2-layer NN is that firm results exist for the former on the selection of the number of hidden layer neurons L for a specified approximation accuracy. A disadvantage of LIP networks is that Barron [Barron 1993] has shown a lower bound on the approximation accuracy.

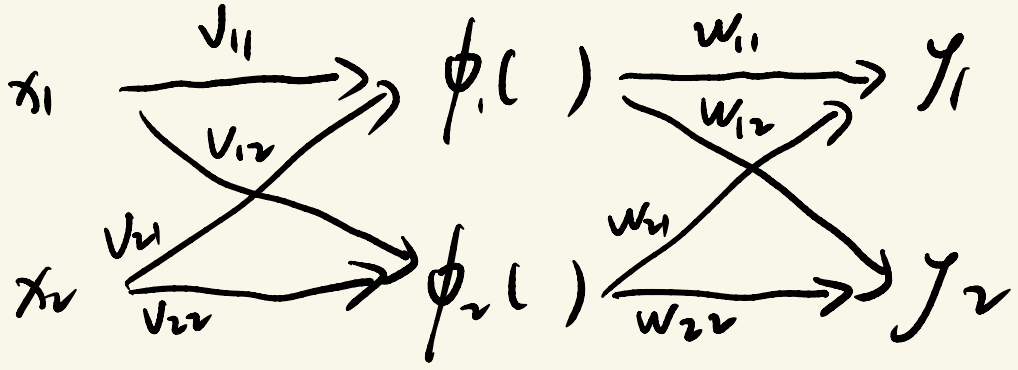
Functional-Link Basis Neural Networks. More generality is gained if $\sigma(\cdot)$ is not diagonal, but $\phi(\cdot)$ is allowed to be a general function from $\Re^n$ to $\Re^L$. This is called a *functional-link neural net (FLNN)* [Sadegh 1993]. For LIP NN, the functional approximation property does not generally hold. However, a 1-layer NN can still approximate functions as long as the activation functions $\phi(\cdot)$ are selected as a *basis*, which must satisfy the following two requirements on a compact simply-connected set of S of $\Re^n$:

1. A constant function on S can be expressed as (8.2.14) for a finite number L of hidden-layer neurons.
2. The functional range of (8.2.14) is dense in the space of continuous functions from S to $\Re^m$ for countable L.

If $\phi(\cdot)$ provides a basis, then a smooth function $f(x)$ from $\Re^n$ to $\Re^m$ can be approximated on a compact set S of $\Re^n$, by

$$f(x) = W^T \phi(x) + \epsilon. \tag{8.2.15}$$

for some ideal weights and thresholds W and some number of hidden layer neurons L. In fact, for any choice of a positive number ϵ_N, one can find a feedforward NN such that $\|\epsilon\| < \epsilon_N$ for all x in S.



$$y_1 = w_{11}\phi_1(\ ) + w_{12}\phi_2(\ )$$

$v_{11}x_1 + v_{21}x_2$ $\qquad$ $v_{12}x_1 + v_{22}x_2$

Value function in Optimal Control

The approximation for $f(x)$ is given by

$$\hat{f}(x) = \hat{W}^T \phi(x), \tag{8.2.16}$$

where W are the NN weight and threshold estimates given by the tuning algorithm. For LIP NN, the functional estimation error is given by

$$\begin{aligned} \tilde{f} &= f - \hat{f} = W^T \phi(x) - \hat{W}^T \phi(x) + \epsilon \\ &= \tilde{W}^T \phi(x) + \epsilon, \end{aligned} \tag{8.2.17}$$

which, when used subsequently in the error dynamics, directly yields tuning algorithms for W that guarantee closed-loop stability.

Barron [Barron 1993] has shown that for all LIP approximators there is a fundamental lower bound, so that ϵ is bounded below by terms on the order of $1/L^{2/n}$. Thus, as the number of NN inputs n increases, increasing L to improve the approximation accuracy becomes less effective. As shown subsequently, this is not a major limitation on adaptive controllers designed using 1-layer NN. This lower bound problem does not occur in the NLIP multilayer nonlinear nets.

A special FLNN is now discussed. We often use $\sigma(\cdot)$ in place of $\phi(\cdot)$, with the understanding that, for LIP nets, this activation function vector is not diagonal, but is a general function from $\Re^L$ to $\Re^L$.

Gaussian or Radial Basis Function (RBF) Networks. The selection of a basis set of activation functions is considerably simplified in various sorts of *structured nonlinear networks,* including radial basis function, CMAC, and fuzzy logic nets. It will be shown here that the key to the design of such structured nonlinear nets lies in a *more general set of NN thresholds* than allowed in the standard equation (8.2.1), and in their appropriate selection.

A NN activation function often used is the gaussian or radial basis function (RBF) [Sanner and Slotine 1991] given when x is a scalar as

$$\sigma(x) = e^{-(x-\bar{x})^2/2p}, \tag{8.2.18}$$

where x is the mean and p the variance. RBF NN can be written as (8.2.4), but have an advantage over the usual sigmoid NN in that the n-dimensional gaussian function is well understood from probability theory, Kalman filtering, and elsewhere, so that n-dimensional RBF are easy to conceptualize.

The j-th activation function can be written as

$$\sigma_j(x) = e^{-\frac{1}{2}(x-\bar{x}_j)^T P_j^{-1}(x-\bar{x}_j)} \tag{8.2.19}$$

with $x, \bar{x}_j \in \Re^n$. Define the vector of activation functions as $\sigma(x) \equiv [\sigma_1(x) \; \sigma_2(x) \ldots \sigma_L(x)]^T$. If the covariance matrix is diagonal so that $P_j = diag\{p_{jk}\}$, then (8.2.19) becomes *separable* and may be decomposed into components as

$$\sigma_j(x) = e^{-\frac{1}{2}\sum_{k=1}^{n} -(x_k-\bar{x}_{jk})^2/p_{jk}} = \prod_{k=1}^{n} e^{-\frac{1}{2}(x_k-\bar{x}_{jk})^2/p_{jk}}, \tag{8.2.20}$$

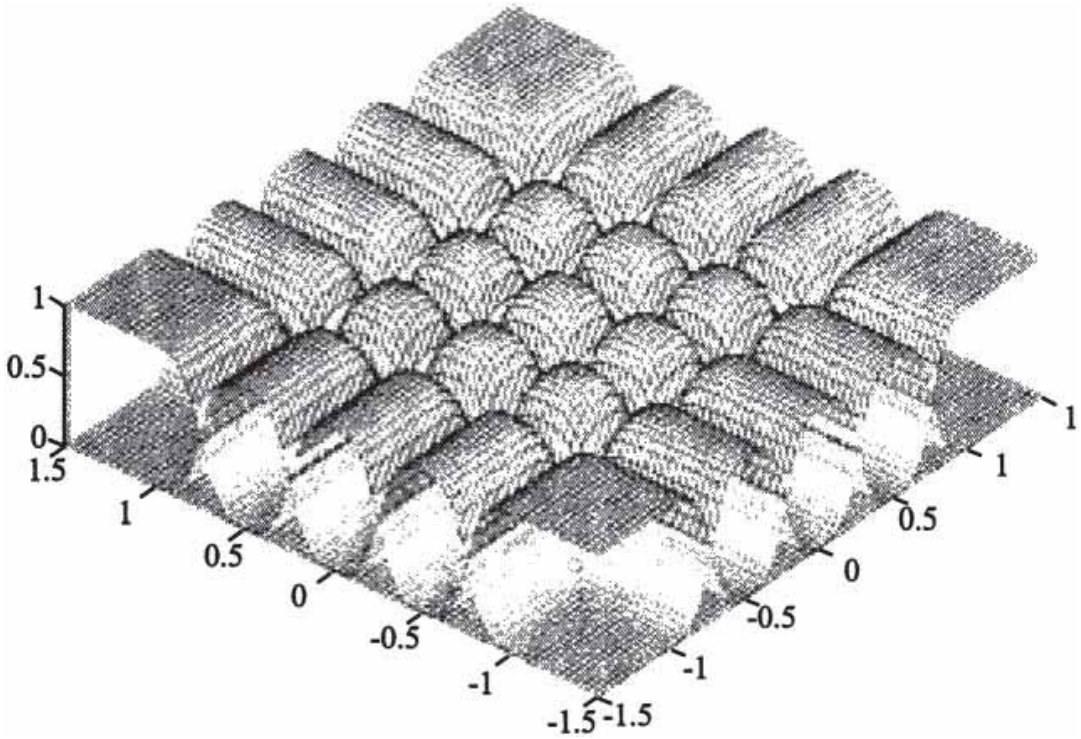


Figure 8.2.2:2-dimensional separable gaussian functions for an RBF NN.

where $x_k, \bar{x}_{jk}$ are the k-th components of $x, \bar{x}_j$. Thus, the n-dimensional activation functions are the product of n scalar functions. Note that this equation is of the form of the activation functions in (8.2.1), but with *more general thresholds*, as a threshold is required for each different component of x at each hidden layer neuron j; that is, the threshold at each hidden-layer neuron in Figure 8.2.1 is a *vector*. The RBF variances p_{jk} and offsets x_{jk} are usually selected in designing the RBF NN and left fixed; only the output-layer weights W^T are generally tuned. Therefore, the RBF NN is a special sort of FLNN (8.2.14) (where $\phi(x) = \alpha(x)$).

Figure 8.2.2 shows separable gaussians for the case $x \in \Re^2$. In this figure, all the variances p_{jk} are identical, and the mean values x_{jk} are chosen in a special way that spaces the activation functions at the node points of a 2-D grid. To form an RBF NN that approximates functions over the region $\{-1<x_1\leq 1, -1<x_2\leq 1\}$ one has here selected $L=5\times 5=25$ hidden-layer neurons, corresponding to 5 cells along x_1 and 5 along x_2. Nine of these neurons have 2-D gaussian activation functions, while those along the boundary require the illustrated 'one-sided' activation functions.

8.3 Tracking Control Using Static Neural Networks

In this section is discussed feedback tracking control design using nonlinear nets and assuming full state-variable feedback. For more details see [Lewis et al. 1999] and other listed references. If full state feedback is available, then a static feedforward NN suffices for control. A control topology and net tuning algorithms are provided here that guarantee closed-loop stability and bounded weights. The techniques to be discussed apply for general nonlinear nets including both neural networks and fuzzy logic systems [Wang 1997], [Lewis et al. 1997], so that the abbreviation NN might henceforth be construed as meaning 'nonlinear network'. The resulting multiloop control topology has an *outer PD tracking loop* and an inner NN feedback linearization or *action generating loop*. It is found that backprop tuning does not generally suffice, but modified tuning algorithms are needed for guaranteed closed-loop performance.

Many industrial mechanical systems, as well as automobiles, aircraft, and spacecraft, have dynamics in the *Lagrangian form,* which are exemplified by the class of rigid robot systems. Therefore, the Lagrangian robot dynamics will be considered [Lewis et al. 1993], [Lewis et al. 1996]. The NN control techniques presented may also be applied to other unknown systems including certain important classes of nonlinear systems [Lewis et al. 1999].

Robot Arm Dynamics and Error System

The dynamics of rigid Lagrangian systems, including robot arms, have some important physical, structural, and passivity properties [Craig 1988], [Lewis et al. 1993], [Slotine 1988], [Spong and Vidyasagar 1989] that make it very natural to use NN in their control. These properties should be taken into account in the design of any controller—in fact, they provide the foundation for rigorous design algorithms for NN controllers.

The dynamics of an n-link rigid (i.e. no flexible links or high-frequency

joint/motor dynamics) robot manipulator may be expressed in the Lagrange form

$$M(q)\ddot{q} + V_m(q,\dot{q})\dot{q} + G(q) + F(\dot{q}) + \tau_d = \tau \tag{8.3.1}$$

with $q(t) \in \Re^n$ the joint variable vector, whose entries are the robot arm joint angles or link extensions. $M(q)$ is the inertia matrix, $Vm(q,q)$ the coriolis/centripetal matrix, $G(q)$ the gravity vector, and $F(q)$ the friction. Bounded unknown disturbances (including e.g. unstructured unmodelled dynamics) are denoted by τ_d, and the control input torque is $\tau(t)$ The robot dynamics have the following standard properties:

Property 1: $M(q)$ is a positive definite symmetric matrix bounded by $m_1 I < M(q) < m_2 I$, with m_1, m_2 positive constants.

Property 2: The norm of the matrix $Vm(q, q)$ is bounded by $v_b(q)\|q\|$, for some function $v_b(q)$.

Property 3: The matrix $M - 2Vm$ is skew-symmetric. This is equivalent to the fact that the internal forces do no work.

Property 4: The unknown disturbance satisfies $\|\tau_d\| < d_B$ with b_D a positive constant.

Given a desired arm trajectory $q_d(t) \in \Re^n$ the tracking error is

$$e(t) = q_d(t) - q(t) \tag{8.3.2}$$

and the filtered tracking error is

$$r = \dot{e} + \Lambda e \tag{8.3.3}$$

where Λ is a symmetric positive definite design parameter matrix, usually selected diagonal. If a controller is found such that $r(t)$ is bounded, then $e(t)$ is also bounded; in fact $\|e\| \leq \|r\|/\lambda_{min}(\Lambda)$ and $\|\dot{e}\| \leq \|r\|$, with $\lambda_{min}(\Lambda)$ the minimum singular value of Λ.

Differentiating $r(t)$ and using (8.3.1), the arm dynamics may be written in terms of the filtered tracking error as

$$M\dot{r} = -V_m r - \tau + f + \tau_d \tag{8.3.4}$$

where the nonlinear robot function is

$$f(x) = M(q)(\ddot{q}_d + \Lambda\dot{e}) + V_m(q,\dot{q})(\dot{q}_d + \Lambda e) + G(q) + F(\dot{q}). \tag{8.3.5}$$

$$M\dot{r} = M\ddot{e} + M\Lambda\dot{e}$$

$$= M(\ddot{q}_d - \ddot{q}) + M\Lambda\dot{e}$$

$$= M\ddot{q}_d \underline{- M\ddot{q}} + M\Lambda\dot{e}$$

$$= M\ddot{q}_d - \tau + V_m\dot{q} + G(q) + F(\dot{q}) + \tau_d + M\Lambda\dot{e}$$

$$= -V_m r - \tau + \tau_d + \underbrace{M\ddot{q}_d + V(\dot{q}_d + \Lambda e) + G(q) + F(\dot{q}) + M\Lambda\dot{e}}_{\triangleq f}$$

$$= -V_m r - \tau + \tau_d + f$$

$$f \triangleq M\ddot{q}_d + V(\dot{q}_d + \Lambda e) + G(q) + F(\dot{q}) + M\Lambda\dot{e}$$

The vector x required to compute $f(x)$ can be defined, for instance, as

$$x \equiv [e^T \ \dot{e}^T \ q_d^T \ \dot{q}_d^T \ \ddot{q}_d^T]^T, \tag{8.3.6}$$

which can be measured. Function $f(x)$ contains potentially unknown robot parameters including pay load mass and complex forms of friction.

A suitable control input for trajectory following is given by the computed-torque-like control

$$\tau = \hat{f} + K_v r - v \tag{8.3.7}$$

with $K_v = K_v^T > 0$ a gain matrix, generally chosen diagonal, and $\hat{f}(x)$ an estimate of the robot function $f(x)$ that is provided by some means. The robustifying signal $v(t)$ is needed to compensate for unmodelled unstructured disturbances. Using this control, the closed-loop error dynamics is

$$M\dot{r} = -(K_v + V_m)r + \tilde{f} + \tau_d + v. \tag{8.3.8}$$

In computing the control signal, the estimate f can be provided by several techniques, including adaptive control or neural or fuzzy networks. The auxiliary control signal $v(t)$ can be selected by several techniques, including sliding-mode methods and others under the general aegis of robust control methods.

The desired trajectory is assumed bounded so that

$$\left\| \begin{matrix} q_d(t) \\ \dot{q}_d(t) \\ \ddot{q}_d(t) \end{matrix} \right\| \leq q_B, \tag{8.3.9}$$

with q_B a known scalar bound. It is easy to show that for each time t, $x(t)$ is bounded by

$$\|x\| \leq c_1 + c_2\|r\| \leq q_B + c_0\|r(0)\| + c_2\|r\| \tag{8.3.10}$$

for computable positive constants c_0, c_1, c_2.

Adaptive Control

Standard adaptive control techniques in robotics [Craig 1988], [Slotine 1988], [Spong and Vidyasagar 1989], [Lewis et al. 1993] use the assumption that the nonlinear robot function is linear in the tunable parameters so that

$$f(x) = \Theta(x)p + \epsilon \tag{8.3.11}$$

where p is a vector of unknown parameters and $\Theta(x)$ is a known *regression matrix*. The control is selected as

$$\tau = K_v r + \Theta(x)\hat{p} \tag{8.3.12}$$

and tuning algorithms are determined for the parameter estimates P. This is facilitated by the LIP assumption. If the approximation error ϵ is non-zero, then the tuning algorithm must be modified in order to guarantee bounded parameter estimates. Various robustifying techniques may be used for this including the σ-mod [Ioannou and Kokotovic 1984], the e-modification [Narendra and Annaswamy 1987], or the dead-zone techniques [Kreisselmeier and Anderson 1986]. There are adaptive control techniques by now that do not require LIP or the determination of a regression matrix [Colbaugh and Seraji 1994]. It is interesting to compare the complexity of these to the NN controllers to be discussed herein.

Neural Net Feedback Tracking Controller

A NN will be used in the control $\tau(t)$ in (8.3.7) to provide the estimate f for the unknown robot function $f(x)$. The NN approximation property assures us that there always exists a NN that can accomplish this within a given accuracy ϵ_N. For 2-layer NLIP NN the approximation is

$$\hat{f}(x) = \hat{W}^T \sigma(\hat{V}^T x), \tag{8.3.13}$$

while for 1-layer LIP NN it is

$$\hat{f}(x) = \hat{W}^T \phi(x), \tag{8.3.14}$$

with $\phi(\cdot)$ selected as a basis.

The structure of the NN controller appears in Figure 8.3.1, where $e \equiv [e^T \quad \dot{e}^T]^T$, $\underline{q} \equiv [q^T \quad \dot{q}^T]^T$. The neural network that provides the estimate for $f(x)$ appears in an *inner control loop,* and there is an *outer tracking loop* provided by the PD term $K_v r$. In control theory terminology, the inner loop is a *feedback linearization* controller [Slotine and Li 1991], while in computer science terms it is the *action generating loop* [Werbos 1992], [Miller et al. 1991], [White and Sofge 1992]. This *multiloop intelligent control structure* is derived naturally from robot control notions, and is not ad hoc. As such, it is immune to philosophical deliberations concerning suitable NN control topologies including the common discussions on feed-forward vs. feedback, direct vs. indirect, and so on. It is to be noted that the static feedforward NN in this diagram is turned into a *dynamic NN* by closing a feedback loop around it (c.f. [Narendra 1991]).

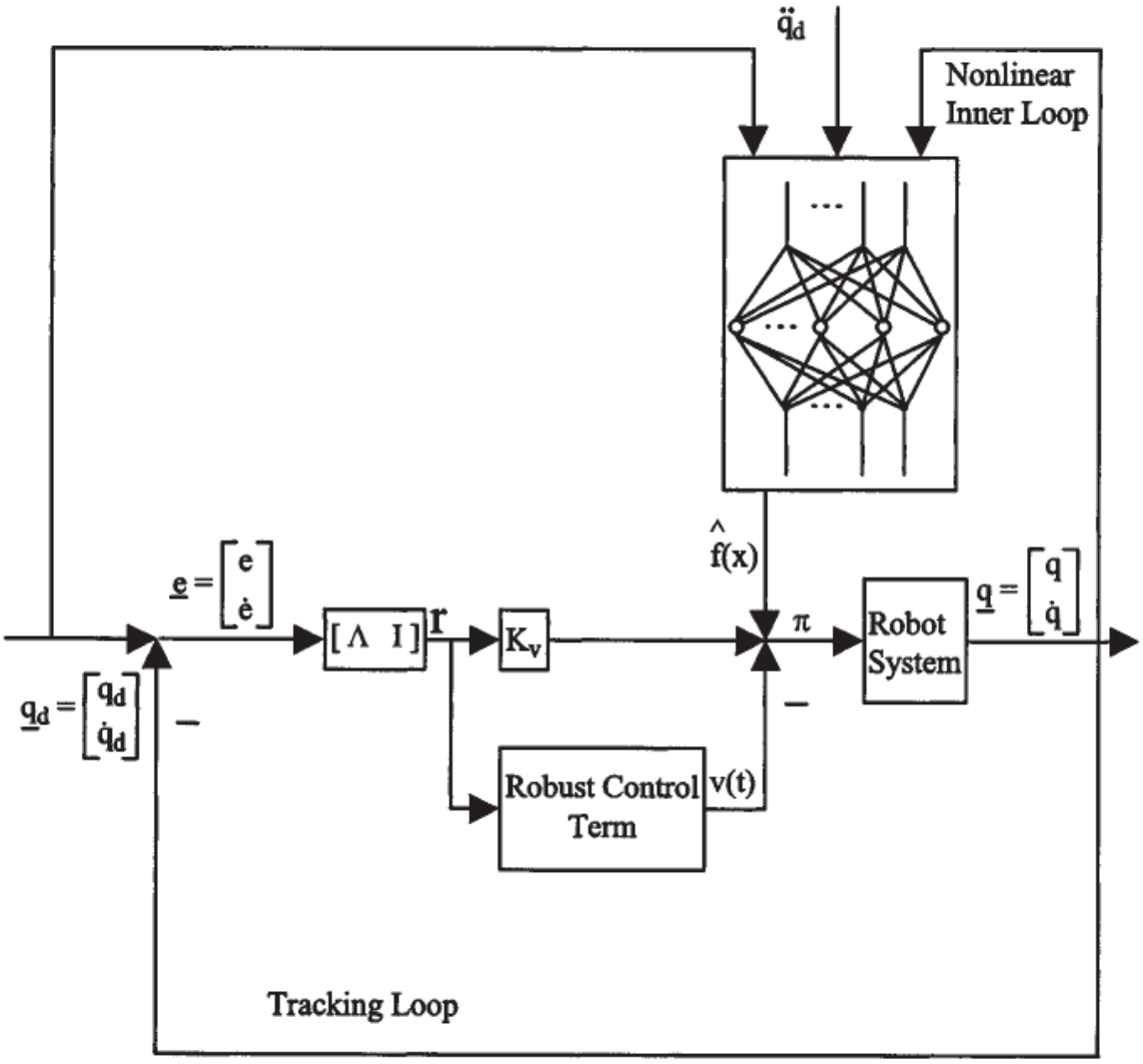


Figure 8.3.1: Neural net robot controller, showing nonlinear neural network loop and outer tracking loop.

Advantage of NN over LIP Adaptive Control. Each robotic system has its own regression matrix, so that a different $\Theta(x)$ must be determined for each system. This regression matrix is often complicated and determining it can be time consuming. One notes that the regression matrix effectively provides a basis for function approximation for a *specific* given system. On the other hand, the universal approximation property of non-linear networks shows that NN provide a basis for *all* sufficiently smooth systems. Therefore, NN can be used to approximate smooth *f(x)* for *all rigid robotic systems,* effectively allowing the design of a single tunable controller for a class of systems. No regression matrix need be determined, for the same NN activation functions suffice for the whole class of plants.

Even the 1-layer NN, which is LIP, has this advantage. Note that the 1-layer NN is linear not in the *system parameters,* but in the NN weights. Even the LIP NN has a universal approximation property so that no regression matrix is needed.

Initial Tracking Errors and Initial NN Weights. It is now required to determine how to tune the NN weights to yield guaranteed closed-loop stability. Several cases are considered for NN controller design in this section. All of them need the following construction.

Since the NN approximation property holds on a compact set, one must define an *allowable initial condition set* as follows. Let the NN approximation property hold for the function $f(x)$ given in (8.3.5) with a given accuracy ϵ_N in (8.2.6) for all x inside the ball of radius $b_x > q_B$. Define the set of allowable initial tracking errors as

$$S_r = \{r \mid \|r\| < (b_x - q_B)/(c_0 + c_2)\}. \tag{8.3.15}$$

Note that the approximation accuracy of the NN determines size of S_r. For a larger NN (i.e. more hidden-layer units), ϵ_N is small for a larger radius b_x. Thus, the allowed initial condition set S_r is larger. On the other hand, a more active desired trajectory (e.g. containing higher frequency components) results in a larger acceleration $q_d(t)$, which yields a larger bound q_B, thereby decreasing S_r. It is important to note the dependence of S_r on the PD design ratio Λ—both c_0 and c_2 depend on Λ.

A key feature of our the Initial Condition Requirement is its *independence of the NN initial weights*. This is in stark contrast to other techniques in the literature where the proofs of stability depend on selecting some initial stabilizing NN weights, which is very difficult to do.

8.4 Tuning Algorithms for Linear-in-the-Parameters NN

Suppose now that a LIP FLNN is used to approximate the nonlinear robot function (8.3.5) according to (8.2.15) with $\|\epsilon\| \leq \epsilon_N$ on a compact set, the ideal approximating weights W constant, and $\phi(x)$ selected as a basis. An estimate of $f(x)$ is given by (8.3.14). Then the control law (8.3.7) becomes

$$\tau = \hat{W}^T \phi(x) + K_v r - v \tag{8.4.1}$$

and the closed-loop filtered error dynamics (8.3.8) are

$$M\dot{r} = -(K_v + V_m)r + \tilde{W}^T \phi(x) + (\epsilon + \tau_d) + v. \tag{8.4.2}$$

$$M\dot{r} = -V_m r - \tau + \tau_d + f$$

Select $\tau = \hat{f} + k_v r - v$

$$M\dot{r} = -(V_m + k_v) r - \hat{f} + f + v + \tau_d$$

$\hat{f} = \hat{w}^T \phi(x)$

$f = w^T \phi(x) + \varepsilon$

$$= -(V_m + k_v) r + (w - \hat{w})^T \phi(x) + \varepsilon + v + \tau_d$$

$$= -(V_m + k_v) r + \tilde{w}^T \phi(x) + \varepsilon + v + \tau_d$$

The next theorem derives the tuning law for a NN controller that is augmented by an *e*-mod term as in [Narendra and Annaswamy 1987].

It must now be assumed that the ideal weights W are constant and *bounded* so that

$$\|W\|_F \leq W_B, \tag{8.4.3}$$

with the bound W_B known. In [Polycarpou 1996] it is shown how to use standard adaptive robust techniques to avoid the assumption that the bound W_B is known.

THEOREM 8.4–1: *(NN Weight Tuning Algorithm).*

Let the desired trajectory $q_d(t)$ be bounded by q_B and the initial tracking error $r(0)$ be in S_r. Let the NN reconstruction error bound ϵ_N and the disturbance bound dB be constants. Assume the ideal NN target weights are bounded by W_B as in (8.4.3). Let the control input for the robot arm be given by (8.4–1) with $v(t)$=0 and gain

$$K_{v_{min}} > \frac{(\kappa W_B^2/4 + \varepsilon_N + d_B)(c_0 + c_2)}{b_x - q_B}. \tag{1}$$

Let the weight tuning be

$$\dot{\hat{W}} = F\phi r^T - \kappa F\|r\|\hat{W}, \tag{2}$$

with F=F^T>0 and κ>0 a small design parameter. Then the filtered tracking error $r(t)$ and the NN weight estimates $W(t)$ are UUB with practical bounds given respectively by the right-hand sides of (8), (9). Moreover, the tracking error may be made as small as desired by increasing the tracking gain K_v.

Proof:

Let the NN approximation property (8.2.15) hold for the function $f(x)$ given in (8.3.5) with a given accuracy ϵ_N for all x in the compact set $S_x \equiv \{x \mid \|x\| < b_x\}$ with b_x>q_B. Let $r(0)\in Sr$. Then the approximation property holds at time 0.

Define the Lyapunov function candidate

$$L = \frac{1}{2}r^T M r^T + \frac{1}{2}tr\{\tilde{W}^T F^{-1}\tilde{W}\} \tag{3}$$

Differentiating yields

$$\dot{L} = r^T M\dot{r} + \frac{1}{2}r^T \dot{M}r + tr\{\tilde{W}^T F^{-1}\dot{\tilde{W}}\} \tag{4}$$

$\dot{\tilde{w}} = \dot{w} - \dot{\hat{w}} = -\dot{\hat{w}}$

whence substitution from (8.4.2) yields

$$\dot{L} = -r^T K_v r + \frac{1}{2} r^T (\dot{M} - 2V_m) r + tr\{\tilde{W}^T (F^{-1}\dot{\tilde{W}} + \phi r^T)\} + r^T(\epsilon + \tau_d). \quad (5)$$

Then, using tuning rule (2) yields

$$\dot{L} = -r^T K_v r + \kappa \|r\| tr\{\tilde{W}^T (W - \tilde{W})\} + r^T(\varepsilon + \tau_d). \quad (6)$$

Since

$$tr\{\tilde{W}^T (W - \tilde{W})\} = < \tilde{W}, W >_F - \|\tilde{W}\|_F^2 \leq \|\tilde{W}\|_F \|W\|_F - \|\tilde{W}\|_F^2,$$

there results

$$\begin{aligned} \dot{L} \leq & - K_{v_{min}} \|r\|^2 + \kappa \|r\| \cdot \|\tilde{W}\|_F (W_B - \|\tilde{W}\|_F) + (\varepsilon_N + d_B)\|r\| \\ = & - \|r\| [K_{v_{min}} \|r\| + \kappa \|\tilde{W}\|_F (\|\tilde{W}\|_F - W_B) - (\varepsilon_N + d_B)], \end{aligned} \quad (7)$$

which is negative as long as the term in braces is positive. Completing the square yields

$$\begin{aligned} & K_{v_{min}} \|r\| + \kappa \|\tilde{W}\|_F (\|\tilde{W}\|_F - W_B) - (\varepsilon_N + d_B) \\ = & \kappa (\|\tilde{W}\|_F - W_B/2)^2 - \kappa W_B^2/4 + K_{v_{min}} \|r\| - (\varepsilon_N + d_B) \end{aligned}$$

which is guaranteed positive as long as

$$\|r\| > \frac{\kappa W_B^2/4 + (\varepsilon_N + d_B)}{K_{v_{min}}} \equiv b_r \quad (8)$$

or

$$\|\tilde{W}\|_F > W_B/2 + \sqrt{\kappa W_B^2/4 + (\varepsilon_N + d_B)/\kappa} \equiv b_W. \quad (9)$$

Thus, L is negative outside a compact set. Selecting the gain according to (1) ensures that the compact set defined by $\|r\| \leq b_r$ is contained in S_r, so that the approximation property holds throughout. This demonstrates the UUB of both $\|r\|$ and $\|W\|_F$. ■

This proof is similar to [Narendra and Annaswamy 1987] but is modified to reflect the fact that the NN approximates only on a compact set. In this result one now requires the bound W_B on the ideal weights W, which appears in the PD gain condition (1). Now, bounds are discovered both for the

tracking error *r(t)* and the weights *W(t)*. Note that no persistence of exictation condition is needed to establish the bounds on *W* with the modified weight tuning algorithm.

The NN weight tuning algorithm (2) consists of a backpropagation through time term plus a new second term. The importance of the κ, term, called the e-modification, is that it adds a quadratric term in $\|\tilde{W}\|_F$ in (7), so that it is possible to establish that *L* is negative outside a compact set in the $(\|r\|, \|\tilde{W}\|_F)$ plane [Narendra and Annaswamy 1987]. The *e*-mod term makes the tuning law *robust* to unmodelled dynamics so that the PE condition is not needed. In [Polycarpou and Ioannou 1991] a projection algorithm is used to keep the NN weights bounded. In [Chen and Khalil 1992], [Chen and Liu 1994] a deadzone technique is employed.

Weight Initialization and On-Line Tuning. In the NN control schemes derived in this paper there is *no preliminary off-line learning phase*. The weights are simply initialized at zero, for then Figure 8.3.1 shows that the controller is just a PD controller. Standard results in the robotics literature [Dawson et al. 1990] show that a PD controller gives bounded errors if K_v is large enough. Therefore, the closed-loop system remains stable until the NN begins to learn. The weights are tuned *on-line in real-time* as the system tracks the desired trajectory. As the NN learns *f(x)*, the tracking performance improves. This is a significant improvement over other NN control techniques where one must find some *initial stabilizing weights*, generally an impossible feat for complex nonlinear systems.

Bounds on the Tracking Error and NN Weight Estimation Errors. The right-hand side of (8) can be taken as a *practical bound on the tracking error* in the sense that *r(t)* will never stray far above it. It is important to note from this equation that the tracking error increases with the NN reconstruction error ϵ_N and robot disturbances d_B, yet *arbitrarily small tracking errors may be achieved* by selecting large gains K_v. This is in spite of Barron's lower bound on ϵ.

Note that the NN weights *W* are not guaranteed to approach the ideal unknown weights *W* that give good approximation of *f(x)*. However, this is of no concern as long as *W*–*W* is bounded, as the proof guarantees. This guarantees bounded control inputs so that the tracking objective can be achieved.

8.5 Tuning Algorithms for Nonlinear-in-the-Parameters NN

Now suppose that a 2-layer NN is used to approximate the robot function according to (8.2.5) with $\|\epsilon\| \leq \epsilon_N$ on a compact set, and the ideal approximating weights W, V constant. Then the control law (8.3.7) becomes

$$\tau = \hat{W}^T \sigma(\hat{V}^T x) + K_v r - v. \tag{8.5.1}$$

The proposed NN control structure is shown in Figure 8.3.1.

The NLIP NN controller is far more powerful than the 1-layer FLNN as it is not necessary to select a basis of activation functions. In effect, the weights V are adjusted on-line to automatically provide a suitable basis. The control is nonlinear in the first-layer weights V, which presents complications in deriving suitable weight tuning laws. Auxiliary signal $v(t)$ is a function to be detailed subsequently that provides robustness in the face of higher-order terms in the Taylor series arising from the NLIP problem.

Using the result (8.2.13) to properly address the NLIP problem, the closed-loop error dynamics (8.3.8) can be written as

$$M\dot{r} = -(K_v + V_m)r + \tilde{W}^T(\hat{\sigma} - \hat{\sigma}'\hat{V}^T x) + \hat{W}^T\hat{\sigma}'\tilde{V}^T x + w + v \tag{8.5.2}$$

where the disturbance terms are

$$w(t) = \tilde{W}^T\hat{\sigma}'V^T x + W^T O(\tilde{V}^T x)^2 + \varepsilon + \tau_d \tag{8.5.3}$$

Define

$$Z \equiv \begin{bmatrix} W & 0 \\ 0 & V \end{bmatrix} \tag{8.5.4}$$

and Z, Z equivalently. It is not difficult to show that

$$\|w(t)\| \leq C_0 + C_1\|\tilde{Z}\|_F + C_2\|\tilde{Z}\|_F\|r\| \tag{8.5.5}$$

with C_i known positive constants.

It is assumed that the ideal weights are bounded so that

$$\|Z\|_F \leq Z_B \tag{8.5.6}$$

with Z_B known. (Standard adaptive robust control techniques can be used to lift the assumption that Z_B is known [Polycarpou 1996].)

The next theorem presents the most powerful NN controller in this chapter.

THEOREM 8.5–1: *(Augmented Backprop Weight Tuning for Multi-layer NN Controller).*

Let the desired trajectory $q_d(t)$ be bounded by q_B and the initial tracking error $r(0)$ be in S_r. Let the ideal target NN weights be bounded by known Z_B. Take the control input for the robot dynamics (8.3.1) as (8.5.1) with PD gain satisfying

$$K_{v_{min}} > \frac{(C_0 + \kappa C_3^2/4)(c_0 + c_2)}{b_x - q_B}, \tag{1}$$

where C_3 is defined in the proof. Let the robustifying term be

$$v(t) = -K_z(\|\hat{Z}\|_F + Z_B)r \tag{2}$$

with gain

$$K_z > C_2. \tag{3}$$

Let NN weight tuning be provided by

$$\dot{\hat{W}} = F\hat{\sigma}r^T - F\hat{\sigma}'\hat{V}^T x r^T - \kappa F\|r\|\hat{W} \tag{4}$$

$$\dot{\hat{V}} = Gx(\hat{\sigma}'^T\hat{W}r)^T - \kappa G\|r\|\hat{V} \tag{5}$$

with any constant matrices $F=F^T>0$, $G=G^T>0$, and $\kappa>0$ a small scalar design parameter. Then the filtered tracking error $r(t)$ and NN weight estimates V, W are UUB, with the bounds given specifically by (10) and (11). Moreover, the tracking error may be kept as small as desired by increasing the gains K_v in (8.5.1).

Proof:

Let the NN approximation property (8.2.5) hold for the function $f(x)$ given in (8.3.5) with a given accuracy ε_N for all x in the compact set $S_x \equiv \{x \mid \|x\| < b_x\}$ with $b_x>q_B$. Let $r(0)\in S_r$. Then the approximation property holds at time 0.

Define the Lyapunov function candidate

$$L = \frac{1}{2}r^T M(q)r + \frac{1}{2}tr\left\{\tilde{W}^T F^{-1}\tilde{W}\right\} + \frac{1}{2}tr\left\{\tilde{V}^T G^{-1}\tilde{V}\right\} \tag{6}$$

Differentiating yields

$$\dot{L} = r^T M\dot{r} + \frac{1}{2}r^T\dot{M}r + tr\{\tilde{W}^T F^{-1}\dot{\tilde{W}}\} + tr\{\tilde{V}^T G^{-1}\dot{\tilde{V}}\}. \tag{7}$$

Substituting now from the error system (8.5.2) yields

$$\begin{aligned}\dot{L} = &- r^T K_v r + \frac{1}{2} r^T (\dot{M} - 2V_m) r \\ &+ tr\{\tilde{W}^T (F^{-1}\dot{\tilde{W}} + \hat{\sigma} r^T - \sigma' \hat{V}^T x r^T)\} \\ &+ tr\{\tilde{V}^T (G^{-1}\dot{\tilde{V}} + x r^T \hat{W}^T \hat{\sigma}')\}\end{aligned} \tag{8}$$

The tuning rules give

$$\begin{aligned}\dot{L} &= - r^T K_v r + \kappa \|r\| tr\{\tilde{W}^T (W - \tilde{W})\} \\ &\quad + \kappa \|r\| tr\{\tilde{V}^T (V - \tilde{V})\} + r^T (w + v) \\ &= - r^T K_v r + \kappa \|r\| tr\{\tilde{Z}^T (Z - \tilde{Z})\} + r^T (w + v).\end{aligned}$$

Since

$$tr\{\tilde{Z}^T (Z - \tilde{Z})\} =< \tilde{Z}, Z > - \|\tilde{Z}\|_F^2 \leq \|\tilde{Z}\|_F \|Z\|_F - \|\tilde{Z}\|_F^2,$$

there results

$$\begin{aligned}\dot{L} \leq &- r^T K_v r + \kappa \|r\| \cdot \|\tilde{Z}\|_F (Z_B - \|\tilde{Z}\|_F) \\ &- K_Z (\|\hat{Z}\|_F + Z_B) \|r\|^2 + \|r\| \cdot \|w\| \\ \leq &- K_{v_{min}} \|r\|^2 + \kappa \|r\| \cdot \|\tilde{Z}\|_F (Z_B - \|\tilde{Z}\|_F) \\ &- K_Z (\|\hat{Z}\|_F + Z_B) \|r\|^2 \\ &+ \|r\| \left[C_0 + C_1 \|\tilde{Z}\|_F + C_2 \|r\| \cdot \|\tilde{Z}\|_F \right] \\ \leq &- \|r\| \left\{ K_{v_{min}} \|r\| - \kappa \cdot \|\tilde{Z}\|_F (Z_B - \|\tilde{Z}\|_F) - C_0 - C_1 \|\tilde{Z}\|_F \right\}\end{aligned}$$

where $K_{v_{min}}$ is the minimum singular value of K_v, and the last inequality holds due to (3).

L is negative as long as the term in braces is positive. Defining $C_3 = Z_B + C_1/\kappa$ and completing the square yields

$$\begin{aligned}&K_{v_{min}} \|r\| - \kappa \cdot \|\tilde{Z}\|_F (Z_B - \|\tilde{Z}\|_F) - C_0 - C_1 \|\tilde{Z}\|_F \\ =&\kappa (\|\tilde{Z}\|_F - C_3/2)^2 + K_{v_{min}} \|r\| - C_0 - \kappa C_3^2/4\end{aligned} \tag{9}$$

which is guaranteed positive as long as either

$$\|r\| > \frac{C_0 + \kappa C_3^2/4}{K_{v_{min}}} \equiv b_r \tag{10}$$

or

$$\|\tilde{Z}\|_F > C_3/2 + \sqrt{C_0/\kappa + C_3^2/4} \equiv b_Z. \tag{11}$$

Thus, L is negative outside a compact set. According to the LaSalle extension [Lewis et al. 1993] this demonstrates the UUB of both $\|r\|$ *and* $\|Z\|_F$ *as long as the control remains valid within this set. However, the PD gain condition (1) shows that the compact set defined by* $\|r\| \leq b_r$ *is contained in* S_r*, so that the approximation property holds throughout.* ■

The first terms in the tuning algorithms (4), (5) are nothing but backpropagation of the tracking error through time. The last terms are the Narendra *e*-mod, and the second term in (4) is a novel second-order term needed due to the NLIP of the NN. The robustifying term $v(t)$ is required also due to the NLIP problem.

The comments appearing after Theorem 8.4.1 are germane here. Specifically, there is no preliminary off-line tuning phase and NN weight tuning occurs on-line in real-time simultaneously with control action. Weight initialization is not an issue in the proof and it is not necessary to provide initial stabilizing weights; the weights are simply initialized so that the NN output is equal to zero, for then the PD loop keeps the system stable until the NN begins to learn. However, practical experiments [Gutierrez and Lewis 1998] show that it is important to initialize the weights suitably. A good choice is to select $V(0)$ randomly and $W(0)$ equal to zero.

In practice it is not necessary to compute the constants c_0, c_1, c_2, C_0, C_1, C_2, Z_B nor determine S_r. The size L of the NN and the PD gains K_v are simply selected large and a simulation is performed. The simulation is then repeated with a different L and K_v. Based on the difference in behavior between the two simulations, L and K_v can be modified.

Straight Backpropagation Weight Tuning. The backprop algorithm (usually in discrete-time form) has been proposed innumerable times in the literature for closed-loop control. It can be shown that if the NN approximation error ϵ is zero, the disturbances are $\tau_d(t)$ zero, and there are no higher-order terms in the Taylor series (8.2.12), then the straight backprop algorithm

$$\begin{aligned} \dot{\hat{W}} &= F\hat{\sigma}r^T \\ \dot{\hat{V}} &= Gx(\hat{\sigma}'^T\hat{W}r)^T \end{aligned} \tag{8.5.7}$$

may be used for closed-loop control instead of (4), (5). PE is required to show that the NN weights are bounded using straight backprop. The conditions mentioned are very restrictive and do not occur in practice; they essentially require the robot arm to be linear (e.g. 1=link). Moreover, PE conditions are not easy to guarantee in NN.

Passivity Properties of NN Controllers

The NN used in this paper are static feedforward nets, but since they appear in a closed feedback loop and are tuned using differential equations, they are dynamical systems. Therefore, one can discuss the *passivity* of these NN. In general a NN cannot be guaranteed to be passive. However, the NN controllers in this paper have some important passivity properties that result in robust closed-loop performance.

The system $x=f(x, u), y=h(x, u)$ is said to be *passive* if it verifies an equality of the so-called *power form* [Slotine and Li 1991]

$$\dot{L}(t) = y^T u - g(t) \tag{8.5.8}$$

for some lower-bounded $L(t)$ and some $g(t) \geq 0$. That is,

$$\int_0^T y^T(\tau)u(\tau)d\tau \geq \int_0^T g(\tau)d\tau - \gamma^2 \tag{8.5.9}$$

for all $T \geq 0$ and some $\gamma \geq 0$. The system is *dissipative* if it is passive and in addition

$$\int_0^\infty y^T(\tau)u(\tau)d\tau \neq 0 \ \text{ implies } \ \int_0^\infty g(\tau)d\tau > 0. \tag{8.5.10}$$

A special sort of dissipativity occurs if $g(t)$ is a quadratic function of $\|x\|$ with bounded coefficients. We call this *state strict passivity (SSP);* then, the norm of the internal states is bounded in terms of the power delivered to the system. Somewhat surprisingly, the concept of SSP has not been extensively used in the literature [Lewis et al. 1993], [Seron et al. 1994], though see [Goodwin et al. 1984] where input and output strict pasivity are defined. SSP turns out to be pivotal in studying the passivity properties of NN controllers, and allows one to conclude some internal boundedness properties without any assumptions of persistence of excitation.

Passivity of the Robot Tracking Error Dynamics

The error dynamics in this paper (e.g. (8.5.2)) appear in Figure 8.5.1 and have the form

$$M\dot{r} = -(K_v + V_m)r + \xi_0 \tag{8.5.11}$$

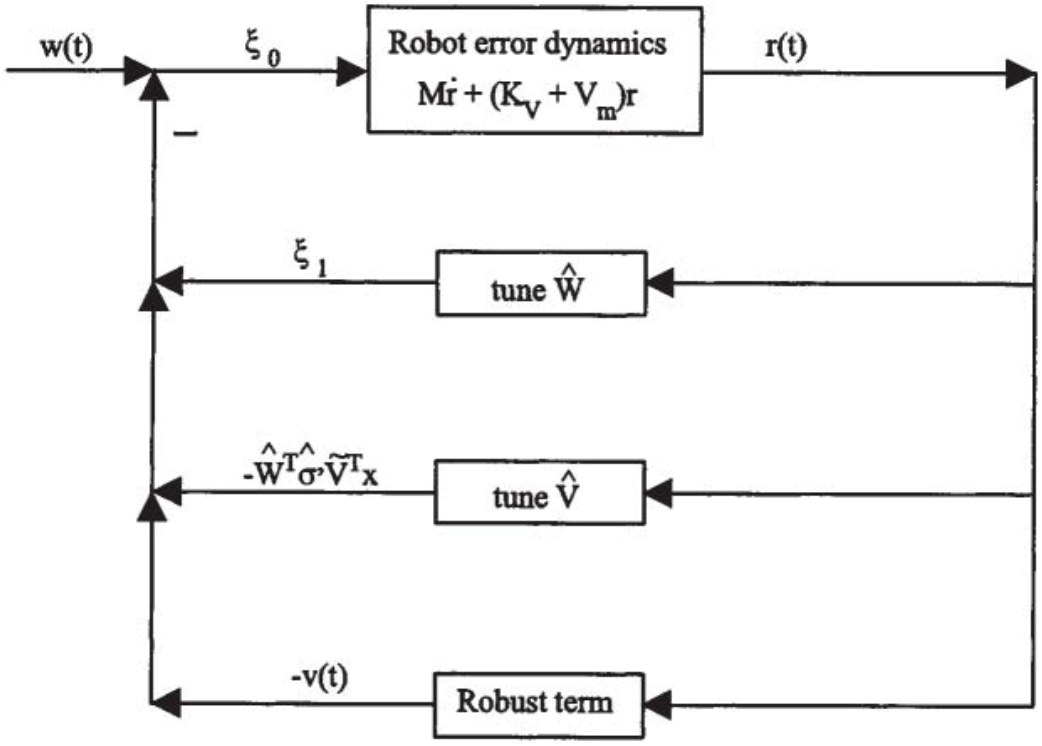


Figure 8.5.1: Two-layer neural net closed-loop error system.

where $r(t)$ is the filtered tracking error and $\xi_0(t)$ is appropriately defined. This system satisfies the following strong passivity property.

THEOREM 8.5–2: *(SSP of Robot Error Dynamics).*

The dynamics (8.5.11) from $\xi_0(t)$ to $r(t)$ are a state strict passive system.

Proof:

Take the nonnegative function

$$L = \frac{1}{2} r^T M(q) r$$

so that, using (8.5.11), one obtains

$$\dot{L} = r^T M \dot{r} + \frac{1}{2} r^T \dot{M} r = -r^T K_v r + \frac{1}{2} r^T (\dot{M} - 2V_m) r + r^T \xi_0$$

whence, the skew-symmetry property yields the power form

$$\dot{L} = r^T \xi_0 - r^T K_v r.$$

This is the power delivered to the system minus a quadratic term in $\|r\|$, *verifying state strict passivity.* ■

Passivity Properties of 2-layer NN Controllers

The closed-loop system is in Figure 8.3.1, where the NN appears in an inner feedback linearization loop. The error dynamics for the 2-layer NN controller are given by (8.5.2). The closed-loop error system appears in Figure 8.5.1, with the signal ξ_1 defined as

$$\xi_1(t) = -\tilde{W}^T(\hat{\sigma} - \hat{\sigma}'\hat{V}^T x) \tag{8.5.12}$$

Note the role of the NN in the error dynamics, where it is decomposed into two effective blocks appearing in a typical feedback configuration.

We now reveal the passivity properties engendered by straight backpropagation tuning (8.5.7). To prove this algorithm, one uses the error dynaqmics in a different form than (8.5.2), so that in Figure 8.5.1 one has $\xi_1(t) = -\tilde{W}^T\hat{\sigma}$.

THEOREM 8.5–3: *(Passivity of Backprop NN Tuning Algorithm).*

The simple backprop weight tuning algorithm (8.5.7) makes the map from $r(t)$ *to* $-\tilde{W}^T\hat{\sigma}$, *and the map from r(t) to* $-\hat{W}^T\hat{\sigma}'\tilde{V}^T x$, *both passive maps.*

Proof:

The dynamics with respect W, V are

$$\dot{\hat{W}} = -F\hat{\sigma}r^T \tag{1}$$

$$\dot{\hat{V}} = -Gx(\hat{\sigma}'^T\hat{W}r)^T \tag{2}$$

1. Selecting the nonnegative function

$$L = \frac{1}{2}tr\{\tilde{W}^T F^{-1}\tilde{W}\}$$

and evaluating L along the trajectories of (1) yields

$$\dot{L} = tr\{\tilde{W}^T F^{-1}\dot{\tilde{W}}\} = -tr\{\tilde{W}^T\hat{\sigma}r^T\} = r^T(-\tilde{W}^T\hat{\sigma}),$$

which is in power form (8.5.8).

2. Selecting the nonnegative function

$$L = \frac{1}{2}tr\{\tilde{V}^T G^{-1}\tilde{V}\}$$

and evaluating along the trajectories of (2) yields

$$\dot{L} = tr\{\tilde{V}^T G^{-1} \dot{\tilde{V}}\} = -tr\{\tilde{V}^T x(\hat{\sigma}'^T \hat{W} r)^T\} = r^T(-\hat{W}^T \hat{\sigma}' \tilde{V}^T x).$$

which is in power form. ■

Thus, the robot error system in Figure 8.5.1 is state strict passive (SSP) and the weight error blocks are passive; this guarantees the passivity of the closed-loop system. Using the passivity theorem [Slotine and Li 1991] one may now conclude that the input/output signals of each block are bounded as long as the external inputs are bounded.

Unfortunately, though passive, the closed-loop system cannot be guaranteed to be SSP, so that when disturbances are nonzero, this does not yield boundedness of the internal states of the weight blocks (i.e. *W, V*) unless those blocks are *observable,* that is persistently exciting (PE).

The next result shows why a PE condition is not needed with the modified weight update algorithm given in Theorem 8.5.1.

THEOREM 8.5–4: *(SSP of Augmented Backprop NN Tuning Algorithm).*

The modified weight tuning algorithms in Theorem 8.5.1 make the map from $r(t)$ *to* $-\tilde{W}^T(\hat{\sigma} - \hat{\sigma}'\hat{V}^T x)$*, and the map from* $r(t)$ *to* $-\hat{W}^T\hat{\sigma}'\tilde{V}^T x$*, both state strict passive (SSP) maps.*

Proof:

The dynamics relative to W, V using the tuning algorithms in Theorem 8.5.1 are given by

$$\dot{\tilde{W}} = -F\hat{\sigma}r^T + F\hat{\sigma}'\hat{V}^T x r^T + \kappa F\|r\|\hat{W}$$

$$\dot{\tilde{V}} = -Gx(\hat{\sigma}'^T\hat{W}r)^T + \kappa G\|r\|\hat{V}.$$

1. Selecting the nonnegative function

$$L = \frac{1}{2}tr\{\tilde{W}^T F^{-1}\tilde{W}\}$$

and evaluating L yields

$$\begin{aligned}\dot{L} &= tr\{\tilde{W}^T F^{-1}\dot{\tilde{W}}\} \\ &= tr\left\{\left[-\tilde{W}^T(\hat{\sigma} - \hat{\sigma}'^T\hat{V}^T x)\right] r^T + \kappa\|r\|\tilde{W}^T\hat{W}\right\}\end{aligned}$$

Since

$$tr\{\tilde{W}^T(W-\tilde{W}) = <\tilde{W},\ W>_F - \|\tilde{W}\|_F^2$$
$$\leq \|\tilde{W}\ \|_F \cdot \|W\|_F - \|\tilde{W}\|_F^2,$$

there results

$$\dot{L} \leq r^T\left[-\tilde{W}^T(\hat{\sigma}-\hat{\sigma}'\hat{V}^Tx)\right] - \kappa\|r\|\cdot(\|\tilde{W}\|_F^2 - \|\tilde{W}\|_F\|W\|_F)$$
$$\leq r^T\left[-\tilde{W}^T(\hat{\sigma}-\hat{\sigma}'\hat{V}^Tx)\right] - \kappa\|r\|\cdot(\|\tilde{W}\|_F^2 - W_B\|\tilde{W}\|_F)$$

which is in power form with the last function quadratic in

2. *Selecting the nonnegative function*

$$L = \frac{1}{2}tr\{\tilde{V}^TG^{-1}\tilde{V}\}$$

and evaluating L yields

$$\dot{L} = tr\{\tilde{V}^TG^{-1}\dot{\tilde{V}}\}$$
$$= r^T(-\hat{W}^T\hat{\sigma}'\tilde{V}^Tx) - \kappa\|r\|(\|\tilde{V}\|_F^2 - <\tilde{V},\ V>_F)$$
$$\leq r^T(-\hat{W}^T\hat{\sigma}'\tilde{V}^Tx) - \kappa\|r\|(\|\tilde{V}\|_F^2 - V_B\|\tilde{V}\|)$$

which is in power form with the last function quadratic in $\|V\|_F$. ■

The SSP of both the robot dynamics and the weight tuning blocks does guarantee SSP of the closed-loop system, so that *the norms of the internal states are bounded in terms of the power delivered to each block*. Then, boundedness of input/output signals assures state boundedness without any sort of observability or PE requirement.

Definition of Passive NN and Robust NN. We define a dynamically tuned NN as *passive* if, in the error formulation, the tuning guarantees the passivity of the weight tuning subsystems. Then, an extra PE condition is needed to guarantee boundedness of the weights. This PE condition is generally in terms of the outputs of the hidden layers of the NN. We define a dynamically tuned NN as *robust* if, in the error formulation, the tuning guarantees the SSP of the weight tuning subsystem. Then, no extra PE condition is needed for boundedness of the weights. Note that (1) SSP of the open-loop plant error system is needed in addition for tracking stability, and (2) the NN passivity properties are dependent on the weight tuning algorithm used.

Passivity Properties of 1-Layer NN Controllers

In a similar fashion, it is shown that the FLNN controller tuning algorithm makes the system passive, so that an additional PE condition is needed to verify internal stability of the NN weights. On the other hand, the augmented tuning algorithm in Theorem 8.4.1 yields SSP, so that no PE is needed.

8.6 Summary

Neural network controller design algorithms were given for a general class of industrial Lagrangian motion systems characterized by the rigid robot arms. The design procedures are based on rigorous nonlinear derivations and stability proofs, and yield a *multiloop intelligent control structure* with NN in some of the loops. NN weight tuning algorithms were given that do not require complicated initialization procedures or any off-line learning phase, work on-line in real-time, and offer guaranteed tracking and bounded NN weights and control signals. The NN controllers given here are *model-free controllers* in that they work for any system in a prescribed class without the need for extensive modeling and preliminary analysis to find a 'regression matrix'. Unlike standard robot adaptive controllers, they do not require linearity in the parameters (see also [Colbaugh and Seraji 1994] which does not need LIP). Proper design allows the NN controllers to avoid requirements such as persistence of excitation and certainty equivalence. NN passivity and robustness properties are defined and studied here.

REFERENCES

[Åström and Wittenmark 1989] K.J.Åström and B.Wittenmark, *Adaptive Control,* Addision Wesley, Reading, MA, 1989.

[Barron 1993] Barren, A.R., "Universal approximation bounds for superpositions of a sigmoidal function," *IEEE Trans. Info. Theory,* vol. 39, no. 3, pp. 930–945, May 1993.

[Chen and Khalil 1992] F.-C.Chen and H.K.Khalil, "Adaptive control of nonlinear systems using neural networks," *Int. J. Control,* vol. 55, no. 6, pp. 1299–1317, 1992.

[Chen and Liu 1994] F.-C.Chen and C.-C.Liu, "Adaptively controlling nonlinear continuous-time systems using multilayer neural networks," *IEEE Trans. Automat. Control,* vol. 39, no. 6, pp. 1306–1310, June 1994.

[Colbaugh and Seraji 1994] R.Colbaugh, H.Seraji, and K.Glass, "A new class of adaptive controllers for robot trajectory tracking," *J. Robotic Systems,* vol. 11, no. 8, pp. 761–772, 1994.

[Craig 1988] Craig, J.J., *Adaptive Control of Mechanical Manipulators,* Reading, MA: Addison-Wesley, 1988.

[Dawson et al. 1990] Dawson, D.M., Z.Qu, F.L.Lewis, and J.F.Dorsey, "Robust control for the tracking of robot motion," *Int. J. Control,* vol. 52, no. 3, pp. 581–595, 1990.

[Goodwin et al. 1984] Goodwin, C.G., and K.S.Sin, *Adaptive Filtering, Prediction, and Control,* Prentice-Hall, New Jersey, 1984.

[Gorinevsky 1995] D.Gorinevsky, "On the persistency of excitation in radial basis function network identificaiton of nonlinear systems," *IEEE Trans. Neural Networks,* vol. 6, no. 5, pp. 1237–1244, Sept. 1995.

[Gutierrez and Lewis 1998] L.B.Gutierrez and F.L.Lewis, "Implementation of a neural net tracking controller for a single flexible link: comparison

with PD and PID controllers," *IEEE Trans. Industrial Electronics,* to appear, April 1998.

[Haykin 1994] S.Haykin, *Neural Networks,* IEEE Press, New York, 1994.

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

[Ioannou and Kokotovic 1984] P.A.Ioannou and P.V.Kokotovic, "Instability analysis and improvement of robustness of adaptive control," *Automatica,* vol. 20, no. 5, pp. 583–594, 1984.

[Kim and Lewis 1998] Y.H.Kim and F.L.Lewis, *High-Level Feedback Control with Neural Networks,* World Scientific, Singapore, 1998.

[Kreisselmeier and Anderson 1986] Kreisselmeier, G., and B.Anderson, "Robust model reference adaptive control," *IEEE Trans. Automat. Control,* vol. AC-31, no. 2, pp. 127–133, Feb. 1986.

[Landau 1979] Y.D.Landau, *Adaptive Control,* Marcel Dekker, Basel, 1979.

[Lewis 1999] F.L.Lewis, "Nonlinear Network Structures for Feedback Control", *Asian Journal of Control,* vol. 1, no. 4, pp.205–228, December 1999.

[Lewis et al. 1993] F.L.Lewis, C.T.Abdallah, and D.M.Dawson, *Control of Robot Manipulators,* New York: Macmillam, 1993.

[Lewis et al. 1999] F.L.Lewis, S.Jagannathan, and A.Yesildirek, *Neural Network Control of Robot Manipulators and Nonlinear Systems,* Taylor and Francis, London, 1999.

[Lewis and Kim 1998] F.L.Lewis and Y.H.Kim, "Neural Networks for Feedback Control," Encyclopedia of Electrical and Electronics Engineering, ed. J.G.Webster, New York: Wiley, 1998. To appear.

[Lewis et al. 1993] Lewis, F.L., K.Liu, and A.Yesildirek, "Neural net robot controller with guaranteed tracking performance," *Proc. Int. Symp. Intelligent Control,* pp. 225–231, Chicago., Aug. 1993.

[Lewis et al. 1995] Lewis, F.L., K.Liu, and A.Yesildirek, "Neural net robot controller with guaranteed tracking performance," *IEEE Trans. Neural Networks,* vol. 6, no. 3, pp. 703–715, 1995.

[Lewis et al. 1997] F.L.Lewis, K.Liu, and S.Commuri, "Neural networks and fuzzy logic systems for robot control," in *Fuzzy Logic and Neural Network Applications,* ed. F.Wang, World Scientific Pub., to appear, 1997.

[Lewis et al. 1996] F.L.Lewis, A.Yesildirek, and K.Liu, "Multilayer neural net robot controller: structure and stability proofs," *IEEE Trans. Neural Networks,* vol. 7, no. 2, pp. 1–12, Mar. 1996.

[Miller et al. 1991] W.T.Miller, R.S.Sutton, P.J.Werbos, ed., *Neural Networks for Control,* Cambridge: MIT Press, 1991.

[Narendra 1991] K.S.Narendra, "Adaptive Control Using Neural Networks," *Neural Networks for Control,* pp 115–142. ed. W.T.Miller, R.S.Sutton, P.J.Werbos, Cambridge: MIT Press, 1991.

[Narendra and Annaswamy 1987] K.S.Narendra and A.M.Annaswamy, "A new adaptive law for robust adaptive control without persistent excitation," *IEEE Trans. Automat. Control,* vol. 32, pp. 134–145, Feb., 1987.

[Narendra and Parthasarathy 1991] Narendra, K.S., and K. Parthasarathy, "Gradient methods for the optimization of dynamical systems containing neural networks," *IEEE Trans. Neural Networks,* vol. 2, no. 2, pp. 252–262, Mar. 1991.

[Polycarpou 1996] M.M.Polycarpou, "Stable adaptive neural control scheme for nonlinear systems," *IEEE Trans. Automat. Control,* vol. 41, no. 3, pp. 447–451, Mar. 1996.

[Polycarpou and Ioannou 1991] M.M.Polycarpou and P.A.Ioannou, "Identification and control using neural network models: design and stability analysis," *Tech. Report 91–09–01,* Dept. Elect. Eng. Sys., Univ. S. Cal., Sept. 1991.

[Rovithakis and Christodoulou 1994] G.A.Rovithakis and M.A. Christodoulou, "Adaptive control of unknown plants using dynamical neural networks," *IEEE Trans. Systems, Man, and Cybernetics,* vol. 24, no. 3, pp. 400–412, Mar. 1994.

[Sadegh 1993] N.Sadegh, "A perceptron network for functional identification and control of nonlinear systems," *IEEE Trans. Neural Networks,* vol. 4, no. 6, pp. 982–988, Nov. 1993.

[Sanner and Slotine 1991] R.M.Sanner and J.-J.E.Slotine, "Stable adaptive control and recursive identification using radial gaussian networks," *Proc. IEEE Conf. Decision and Control,* Brighton, 1991.

[Seron et al. 1994] Seron, M.M., D.J.Hill, and A.L.Fradkov, "Adaptive passification of nonlinear systems," *Proc. IEEE Conf. Decision and Control,* pp. 190–195, Dec. 1994.

[Slotine 1988] Slotine, J.-J.E., "Putting physics in control the example of robotics," *IEEE Control Systems Magazine,* pp. 12–17, Dec. 1988.

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

[Spong and Vidyasagar 1989] Spong, M.W., and M.Vidyasagar, *Robot Dynamics and Control,* New York: Wiley, 1989.