Pergamon

Mathl. Comput. Modelling Vol. 27, No. 3, pp. 65–92, 1998

0895-7177/98 $19.00 + 0.00

PII: S0895-7177(97)00266-5

Neurocontrol of Nonlinear Systems via Local Memory Neurons*

S. M. AMIN AND E. Y. RODIN
Center for Optimization and Semantic Control
Department of Systems Science and Mathematics
Washington University in St. Louis
Campus Box 1040, One Brookings Drive
St. Louis, MO 63130-4899, U.S.A.
http://rodin.wustl.edu

A. Y. WU
Compuserve Co., 1901 South Bascom Ave.
Campbell, CA 95008, U.S.A.

(Received May 1997; accepted June 1997)

Abstract—In this report, we consider the part of our work which concerns the approximation of nonlinear dynamic systems using neural networks. Based on a new paradigm of neurons with local memory (NNLM), we discuss the representation of control systems by neural networks. Using this formulation, the basic issues of controllability and observability for the dynamic system are addressed. A separation principle of learning and control is presented for NNLM, showing that the weights of the network do not affect its dynamics. Theoretical issues concerning local linearization via a coordinate transformation and nonlinear feedback are discussed. For illustration of the approach simulation results for nonlinear control of an aircraft encountering wind shear on take-off is presented.

Keywords—Neurocontrol, Dynamic neural networks, Nonlinear control, Aircraft control.

1. INTRODUCTION

Theoretical and practical applications of multilayered artificial neural networks (ANNs) have experienced an enormous revival in recent years; their use for approximation and modeling of "static" systems has been extensively studied. From a theoretical point of view, it has been proved that even with one hidden layer, ANNs with an appropriately chosen number of units can approximate any continuous function over a compact domain [10–12].

Although mathematical theories for linear time-invariant systems are well understood, a comparable overall theory for nonlinear systems is not available. Nonlinear systems are studied on a system-by-system basis (cf. [13,14]). In recent years a number of authors have addressed issues such as controllability, observability, feedback stabilization, and observer design (cf. [15–19]). In spite of such attempts, general constructive procedures similar to those available for linear systems have not been achieved for nonlinear systems. Considerable progress in nonlinear system theory will be needed to obtain rigorous solutions to the identification and control problems.

*The work of the authors affiliated with the Center for Optimization and Semantic Control was supported in part by AFOSR under Grants No. 890158, F49 620-93-01-0012, and F49 620-96-1-0151. Earlier versions of this work have been reported in references [1–9].

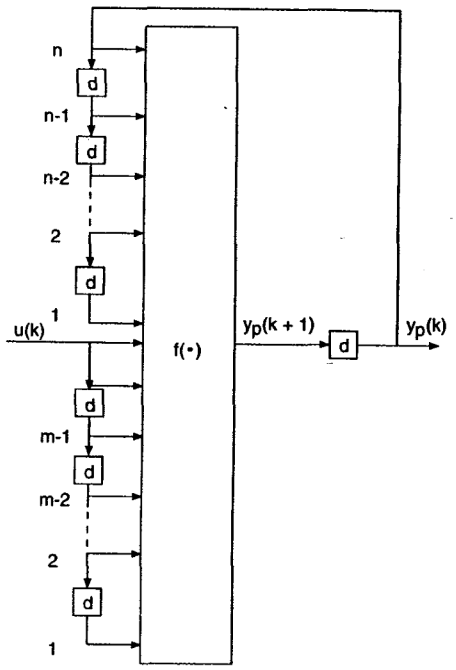


Figure 1. Representation of general SISO discrete-time dynamical system. The term d corresponds to a unit delay (z^{-1}).

In comparison, nonlinear relationships will be learned with relative ease by ANN if sufficiently "interesting" measured data and enough computer power are available. As a result of the fairly fast development of powerful parallel computing systems in the last years, the drawbacks of the latter in the applications of ANN have been progressively decreasing. The significance of ANN concerning nonlinear control systems has been demonstrated by recent research publications, e.g., the two collections [18,19], and briefly reviewed in [20]. Since 1990, a few papers have not only demonstrated satisfactory results in applying approaches of neurocontrol, but also related the theories of classical and modern control systems to ANNs. Fundamental issues such as systems approximation and identification, controllability, observability, and stability theory have been addressed. While major results in approximation and identification have already been established, the latter issues are still in a very early stage. Indeed, the lack of rigorous mathematical representations of neurocontrol systems is a main drawback for the further development of the research in neurocontrol. The most significant advances in the development of a systematic body of transparent and constructive design principles have been made in Neural Adaptive Control Systems (NACS). These were recently reviewed in [21].

ANNs are viewed as a subset of mathematical algorithms, not an alternative to them; they allow development of data-based models permitting system development based on learning instead of tweeking. Such models have enormous potential as building blocks in tomorrow's computational world. Data-based systems show great promise for solving problems that require learning and adaptation during use. Many ANN applications are motivated by new universal approximation theorems showing that some ANNs work better than Taylor series and regression analysis.

More recent developments include the use of ANNs for identification and approximation of dynamical systems [1–12,22]. These networks contain dynamic elements either in the form of neuron output feedback (recurrent neurons) or inclusion of an internal memory which preserves the state history of the unit. The use of such local memories becomes necessary for identification of dynamical systems with higher, unknown orders [22]. Stability issues of such networks are discussed in [23,24]; their superior storage capacity has been investigated in [25,26].

Farotimi *et al.* [27] provided a weight synthesis technique based on optimal control theory for dynamic neural networks. Gori *et al.* [28] presented a backpropagation (BP) algorithm for a particular class of dynamic neural networks where some processing elements have a local feedback, and applied this class of neural networks to the problem of speech recognition. Gherrity [29] derived a learning algorithm for a recurrent neural network which is described by a system of coupled differential equations. Willis *et al.* [30] discussed advantages of a neural network estimator for feedback process control.

Perfetti [31] considered the effect of positive self-feedback on neural networks, and showed that binary output can be guaranteed with finite sigmoid slope such that nonbinary solutions become unstable. Sudharsanan and Sundareshan [32] proposed a descent procedure as a learning rule to minimize the error between the stable equilibrium points of the network and the desired memory vectors. Sato *et al.* [33] discuss their work on an adaptive nonlinear pair oscillator with local connections used for speech synthesis. See [34] for a discussion of stability issues for asymmetric dynamic neural networks. Mckelvey [35] presents a method for developing controllers for nonlinear systems based on an optimal control formulation. The network was trained with the back propagation algorithm where examples of optimal controls previously calculated with a differential dynamic programming technique were used. Werbos [36] and Yamaguchi [37] present overviews of neurocontrol and fuzzy logic formulations for aerospace applications. For collections of papers on neurocontrol see [18,19]; the important properties of ANNs in control are:

(1) their ability to approximate arbitrary nonlinear functions;
(2) parallel processing which provides fault tolerance and can be implemented in hardware;
(3) capability to learn and adapt to a changing environment;
(4) perform data fusion on both quantitative and qualitative data; and
(5) ready application to multivariable systems.

One appealing feature is the potential of neural nets to provide a generic "blackbox" representation for nonlinear models; this has led to investigation of the approximation capabilities of neural nets. An important question is that of system identifiability [38]; i.e., can a system be "adequately" represented via a given model structure?

To answer this question, several researchers have developed theories and algorithms based on adaptive control system identification. For dynamic plants a delay line is used and past output values of the plant are provided as inputs to the neuroidentifier. If the static and dynamic effects are inseparable, a general neural network model is used where past inputs as well as past outputs are fed as inputs to the neuroidentifier. The number of delays can be obtained either from the characteristics of the plant (e.g., order of the transfer function, if available) or from prior knowledge, experimentation, or physics of the problem [39–41]. Most of these approaches are based on replacing the traditional systems identification/controller described for example in adaptive control literature [38,41–43] with a neuroidentifier or neurocontroller.

In [40], Billings *et al.* considered the identification of nonlinear systems. A feedback neural network was trained to model the unknown nonlinear system, i.e., to minimize the error between the output of the network and the output of the actual system. The past inputs as well as the past outputs were fed through a delay line (Figure 1). Several topics were discussed in [40]. They are:

(i) network complexity;
(ii) node selection;
(iii) prediction and the effects of noise;
(iv) biasedness; and
(v) model validation test.

Thorough discussion and numerical simulations are given regarding these issues. It has been shown that while it is easy to train a neural network which predicts well over the estimation set, this does not necessarily mean that the network provides an adequate description of the

underlying mechanism which generated the data. The authors claimed that it was difficult to get definite analytical results in this area, but the modal validity tests introduced there did appear to provide a useful metric of the network performance. An interesting observation of their approach is that the inputs to the neural network can contain lagged external inputs and lagged outputs of the neural network. In fact, without using lagged external inputs and lagged outputs of the network as inputs to the neural networks, the performance of the network would deteriorate.

In [41], four models of discrete-time plants were used in the identification problem. They are as follows.

MODEL I.

$$y_p(k+1) = \sum_{i=0}^{n-1} a_i y_p(k-i).$$

MODEL II.

$$y_p(k+1) = f[y_p(k), y_p(k-1), \ldots, y_p(k-n+1)] + \sum_{i=0}^{m-1} \beta_i u(k-i).$$

MODEL III.

$$y_p(k+1) = f[y_p(k), y_p(k-1), \ldots, y_p(k-n+1)] + g[u(k), u(k-1), \ldots, u(k-m+1)].$$

MODEL IV.

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

where $[u(k), y_p(k)]$ represents the input-output pair of the single-input/single output (SISO) plant at time k, α_i, β_i are unknown model parameters, and $f(.)$ and $g(.)$ are differential functions (in Model II and III, $f : \mathbb{R}^n \to \mathbb{R}$, and in Model IV, $f : \mathbb{R}^{n+m} \to \mathbb{R}$, and $g : \mathbb{R}^m \to \mathbb{R}$). It is not difficult to show that Models I, II, and III are special cases of Model IV. Model IV is the analytical representation of the neural network in Figure 1.

In our approach the "internal information" of the network is parameterized as a control system; the goal has been to represent the identifier and the controller in terms of this information and thus integrate two types of dynamic neural network architectures into controllers. Suitability of feedforward architectures with dynamic neurons for identification and control of dynamic systems has been shown; several important issues concerning controllability, observability, and feedback linearizability of such neural models are also investigated.

2. PRELIMINARIES

2.1. Approximation via Feedforward Artificial Neural Networks (FANN)

FANNs can approximate the continuous mapping f: $\mathcal{R}^{m+n} \to \mathcal{R}$ of an input-output discrete-time description of the nonlinear system (2.1). The representation of a general SISO plant is depicted in Figure 1. Similarly a continuous feedback control law can be approximated.

$$y(k+1) = f[y(k), y(k-1), \ldots, y(k-n+1);\ u(k), u(k-1), \ldots, u(k-m+1)]. \tag{2.1}$$

Approximation theory for ANNs states conditions under which a parameterization of a network guarantees the existence of a uniform approximation of the function f.

Generally FANN are known as a compact mapping (2.2) between two information domains where p is the number of input and q is the number of output patterns. In the following important existence theorems of a uniform approximation of f on $\mathcal{K}$ are presented.

$$\mathcal{K} \to \mathcal{R}, \qquad f \in C^0, \qquad \mathcal{K} \subset \mathcal{R}^{pm+qn}. \tag{2.2}$$

The Weierstrass' Theorem says that a function $f : [a, b] \to \mathcal{R}$ can be uniformly approximated by a sequence of polynomial $\{p_n(x)\}$. The Stone-Weierstrass Theorem states that the general properties of approximating functions are not intrinsic to polynomials.

THEOREM 1. KOLMOGOROV'S THEOREM. *Any function continuous on the unit n-dimensional cube* $E^n (E = [0,1])$ *can be represented in the form*

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{2_n+1} \chi_i \left(\sum_{j=1}^{n} \phi_{ij}(x_j) \right), \tag{2.3}$$

it where χ_i *and* ϕ_{ij} *are real* C° *functions of one variable.*

Kolmogorov's Representation Theorem guarantees the exact representation of every continuous function as a superposition of a finite number of continuous functions of one variable. Hecht-Nielsen applied Theorem 1 to FANN and stated Kolmogorov's Mapping FANN Existence Theorem (cf. [44]).

THEOREM 2. HECHT-NIELSEN'S THEOREM. *Given any continuous function* $f : [0,1]^n \to \mathcal{R}^m$, $f(x) = y$, *f can be implemented exactly by a two-layer*[1] *FANN having n units in the input layer,* $(2n+1)$ *units in the hidden layer and m units in the output layer.*

In contrast to the existence theorems presented so far, Kůrková in [45] introduced an approximation version of Kolmogorov's Theorem. She determined the form of the functions used in the ANN, and furthermore proved that any continuous function can be approximated arbitrarily well by a three-layer FANN with sigmoidal transfer functions.

THEOREM 3. KŮRKOVÁ'S THEOREM. *Let* $n \in \mathcal{N}$ *with* $n \geq 2$, $\sigma : \mathcal{R} \to \mathcal{E}$ *be a sigmoidal function,* $f \in C^0(E^n)$ *and* $\varepsilon \in \mathcal{R}$. *Then* $\exists k \in \mathcal{N}$ *and staircase-like functions* $\chi_i, \phi_{ij} \in \mathcal{S}(\sigma)$ *such that* $\forall (x_1, \ldots, x_2) \in E$

$$\left| f(x_1, \ldots, x_n) - \sum_{i=1}^{k} \chi_i \left(\sum_{j=1}^{n} \phi_{ij}(x_j) \right) \right| < \varepsilon, \tag{2.4}$$

where $\mathcal{S}(\sigma)$ *is the set of all staircase-like functions of the form* $\sum_{i=1}^{k} a_i \sigma(b_i x + c_i)$.

In addition to, the problem of the existence of an approximation of f the problem exists of interpolation of the continuum $f(\mathcal{K})$ from a finite number of sample pairs $(U_k, Y_k) \in \mathcal{K} \times \mathcal{R}^q$, $k = 1, \ldots, s$. Refer to [13] for a brief discussion of this. Basic FANN models are the sigmoidal model (2.5) and the Radial Basis Function (RBF) network (2.6). According to the Stone-Weierstrass Theorem both network structures are suitable for uniform approximation of an arbitrary continuous mapping.

$$\begin{aligned} y_i &= \sum_{j=1}^{N} a_{ij} \sigma \left(b_{ij}^{\top} U + d_{ij} \right), \qquad i = 1, \ldots, q, \quad a_{ij}, b_{ij} \in \mathcal{R}, \quad b_{ij} \in \mathcal{R}^{pm+qn}, \\ &= [y(k-1), \ldots, y(k-n);\ u(k-1), \ldots, u(k-m)]^{\top} \end{aligned} \tag{2.5}$$

$$y_i = \sum_{j=1}^{n} a_{ij} g \left(\frac{\|u - c_{ij}\|}{m_{ij}} \right), \qquad i = 1, \ldots, q. \tag{2.6}$$

The above theorems can be summed up as follows: suppose we want to map any real n-dimensional vector to any other real vector of dimension m; the only constraint we will place on the mapping is that the components of the input vector will all be scaled to lie in the closed unit interval; there is no such constraint on the output vector. Theorems 1 and 2 state that any continuous function of n variables can be computed using only linear summations and nonlinear (but continuously increasing) functions of only one variable; and that a three-layer FANN with continuously increasing nonlinearities can compute any continuous function of n variables. Theorem 2

[1]The number of layers of a ANN refers to the number of layers of weights.

states that a multilayer FANN exists which, instead of providing an approximation, can perform exact mapping. Furthermore, the network will have n neurons in its input layer, m neurons in its output layer, and $2n + 1$ neurons in its middle layer. Theorem 3 extends these results via using sigmoidal nonlinearities. However, the above theorems do not indicate how weights in the network should be selected or how sensitive the output function is to variations in the weights. There are several mechanisms for automating the learning algorithm for adjustment of weights in the multilayered networks. The learning mechanisms can occur in either a *supervised* (desired outputs are known, and are included in the learning rule) or an *unsupervised* (network is not told *a priori* what to learn; it must instead discover similarities among the input patterns) fashion. Back propagation is a supervised learning rule which feeds the error back through the network after every complete presentation of training input patterns. Typically, the transfer function for the two hidden layers and the output layer is sigmoidal; the learning strategy for the training process utilizes back propagation.

The back propagation algorithm adapts the weights using a recursive procedure starting at the output nodes and working back to the first hidden layer. The weights are adjusted by

$$\omega_{ij}(t+1) = \omega_{ij}(t) + \eta\delta_j x_i,$$

where $\omega_{ij}(t)$ is the weight from a hidden node (or an input node) i to a node j at time t; x_i is either the output of node i or an input; η is the gain term (learning rate) between 0 and 1, which represents the speed of convergence; and δ_j is an error term for node j. If node j is an output node with an actual output y_j; then

$$\delta_j = y_j(1 - y_j)(d_j - y_j),$$

where d_j is the desired output of node j. If node j is a hidden node, then

$$\delta_j = x_j(1 - x_j)\sum_k(\delta_k\omega_{jk}),$$

where k is over all nodes in the layers above node j. Convergence may become faster if a momentum coefficient α is added, and weight changes are given by

$$\omega_{ij}(t+1) = \omega_{ij}(t) + \eta\delta_j x_i + \alpha[\omega_{ij}(t) - \omega_{ij}(t-1)].$$

The BP algorithm is perhaps the most widely utilized technique for training FANNs to approximate static functions. As pointed out earlier, recurrent (dynamic) NN structures are required when approximating dynamical systems.

3. NETWORKS WITH LOCAL MEMORY NEURONS (NNLM)

A mathematical representation of an input-output relationship for a neuron can be written as

$$y_k^j = f^j\left(u_k^{1,j}, \ldots, u_k^{n_j,j}\right), \qquad k \in Z, \tag{3.1}$$

where the y's and u's are the outputs and the inputs respectively. Also, Z is the set of positive integers, the subscript k denotes the time step k, and the superscript j denotes the j^{th} neuron. A typical form for f^j of (3.1) can be written as

$$y_k^j = s_j\left(\sum_{i=1}^{n_j} w_{ij}u_k^{ij}\right), \tag{3.2}$$

where s_j is a sigmoidal function, and the w_{ij}'s are the synaptic weights.

The quantities $y_k^j, u_k^{1,j}, \ldots, u_k^{n_j,j}$ are the outputs of and inputs to the neuron at time step k, respectively. Also, z^{-1} denotes the backshift operator and s_j^{-1} denotes the inverse of the activation function for the neuron j. The output y_k^j of a neuron with local memory (NLM) can be written as

$$y_k^j = s_j \left(a^j \, s_j^{-1}(y_{k-1}^j) + c^j \sum_{i=1}^{n_j} w_{ij} \, u_k^{i,j} \right), \tag{3.3}$$

where a^j is a scalar whose value represents the dynamics in neuron j, c^j is another scalar, and the w_{ij}'s are the weights of connection from other neurons to neuron j. By setting $a^j = 0$ and $c^j = 1$, we immediately obtain the conventional input-output relationship for the McCulloch-Pitts neurons. It follows that the input-output relationship of a conventional neuron is actually a special case of that of an NLM.

An alternative and more informative input-output representation of an NLM can be given by introducing an internal state variable x_k

$$x_k^j = a^j \, x_{k-1}^j + \sum_{i=1}^{n_j} w_{ij} \, u_k^{i,j}, y_k^j = s_j \left(c^j x_k^j \right), \qquad k \in Z, \tag{3.4}$$

from which (3.3) can be derived easily. The system equation (3.4) is called the node system. Again setting $a^j = 0$ and $c^j = 1$ in (3.4), we obtain the input-output relationship of a conventional static neuron.

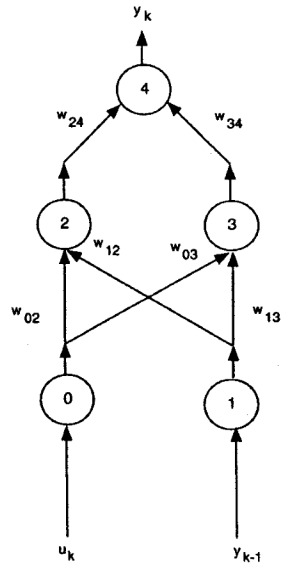


Figure 2. A simple special case of neural networks used for identification.

The advantage of the representation (3.4) over the representation (3.3) is apparent by introducing the internal state x_k^j. The system (3.4) actually has the standard state equation and output equation familiar to control scientists. For convenience, we still adopt the same name, "state equation", for the x equation. The role of a^j in (3.4) is clear from the familiar control theory. For example, a necessary condition for the node system to be asymptotically stable is that the a^j's lie inside the unit disc in the complex plane. Even though the state equation in (3.4) is linear and time-invariant, the output equation is nonlinear, which complicates further analysis. Although we may assume that s_j is linear, which is the case in part of our following analysis, we shall generally consider s_j to be nonlinear, e.g., a commonly used sigmoidal function.

Let us now examine a simple special case of the network shown in Figure 2. The network has two inputs, one hidden layer with two hidden neurons and one output neuron.

The input-output relationship of the network shown in Figure 2 is described by

$$y_k = s_2[w_{24}\, s_1(w_{02}\, u_k + w_{12}\, y_{k-1}) + w_{34}\, s_1(w_{03}\, u_k + w_{13}\, y_{k-1})], \tag{3.5}$$

where $s_1(.)$ is the activation function for the hidden layer and $s_2(.)$ is the activation function for the output neuron. We have assumed that the activation functions for the input neurons are linear. If we further assume that the function $s_1(.)$ is linear, then (3.5) becomes

$$y_k = s_2[w_u\, u_k + w_y\, y_{k-1}], \tag{3.6}$$

where $w_u = w_{24}\, w_{02} + w_{34}\, w_{03}$, $w_y = w_{24}\, w_{12} + w_{34}\, w_{13}$. Next, let us assume that the number of inputs to the NLM in equation (3.3) is one for simplicity. Then, we have the following equation for this special NLM

$$y_k = s\left(as^{-1}(y_{k-1}) + cwu_k\right), \tag{3.7}$$

where we only keep the subscript k and have dropped all other superscripts and subscripts for simplicity. Comparing equation (3.7) with equation (3.3), we readily see the similarity of our models to those in [40,41]. The common point for these representations is: *the past history information of the network is used.* This is because the dynamics are incorporated in the network.

Having defined the basic structure for an NLM, we can now construct a neural network whose elements are NLMs. We shall denote the NNLM with m inputs, n hidden nodes and p outputs by $N_{m,n,p}$. For simplicity, we only consider the single-input and single-output (SISO) system in this section. The generalization to the multi-input and multioutput (MIMO) system is straightforward. Meanwhile, the input to the network has generally arbitrary values.

A general structure for NNLM is shown in Figure 3. The state equations are

$$\begin{aligned}
\text{node } 0: &\begin{cases} x_k^0 &= a^0 x_{k-1}^0 + u_k, \\ y_k^0 &= s_0\left(c^0 x_k^0\right), \end{cases} \\
\text{node } 1, \ldots, \text{node } n-2: &\begin{cases} x_k^i &= a^i x_{k-1}^i + w_{1i}\, y_k^0, \\ y_k^i &= s_2\left(c^i x_k^i\right), \qquad i = 1, 2, \ldots, n-2, \end{cases} \\
\text{node } n-1: &\begin{cases} x_k^{n-1} &= a^{n-1} x_{k-1}^{n-1} + \sum_{i=1}^{n-2} w_{2i}\, y_k^i, \\ y_k &= s_3\left(c^{n-1} x_k^{n-1}\right), \end{cases}
\end{aligned}$$

where the $a^i s$ are scalars representing the dynamics of the i^{th} node system, the s_j's are the activation functions, which are generally sigmoidal functions, and the w_{ij}'s are the synaptic weights for the path connecting adjacent layers.

Assuming for a moment that the activation functions s_0, s_2, and s_3 are all linear, and defining the state-variable vector $\mathbf{x}_k^\top$ by $\mathbf{x}_k^\top = [x_k^0, \ldots, x_k^{n-1}]$, we can represent the node system in a more concise form by

$$\begin{aligned}
\mathbf{x}_k &= \mathbf{A}x_{k-1} + \mathbf{B}u_k, \\
y_k &= \mathbf{C}\mathbf{x}_k,
\end{aligned} \tag{3.8}$$

where

$$\mathbf{A} = \begin{bmatrix}
a^0 & 0 & 0 & \cdots & 0 & 0 \\
w_{11}\, c^0\, a^0 & a^1 & 0 & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
w_{1(n-2)}\, c^0\, a^0 & 0 & 0 & \cdots & a^{n-2} & 0 \\
a^0 \tilde{a} & w_{21}\, c^1\, a^1 & w_{22}\, c^2\, a^2 & \cdots & w_{2(n-2)}\, c^{n-2}\, a^{n-2} & a^{n-1}
\end{bmatrix},$$

$$\mathbf{B}^\top = [1 \quad w_{11} c^0 \quad \cdots \quad w_{1(n-2)}\, c^0 \quad \tilde{a}],$$

$$\mathbf{C} = [0 \quad 0 \quad \cdots \quad 0 \quad c^{n-1}],$$

$$\tilde{a} = \sum_{i=1}^{n-2} w_{1i}\, w_{2i}\, c^0\, c^i.$$

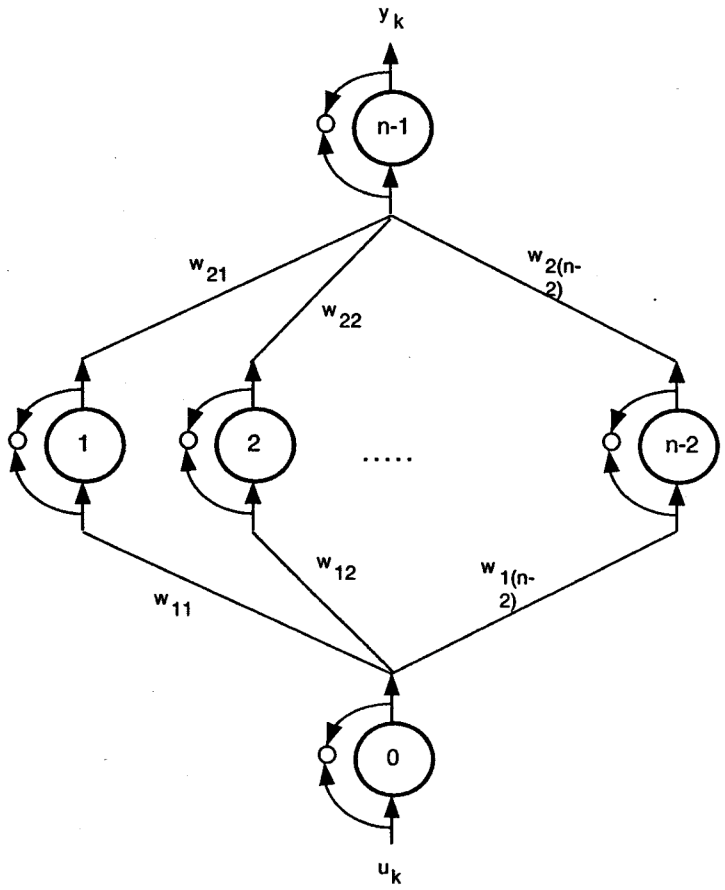


Figure 3. General structure of NNLM.

Equation (3.8) represents a linear state and output equation with the transfer matrix being a lower-triangular one. By assigning a^i (for $0 \leq i \leq n-1$) in A, we can alter the dynamics in (3.8). Assuming that $a^{n-1} \neq a^i$ for $i = 0, \ldots, n-2$, we define the quantity a_c as follows:

$$a_c = \sum_{i=1}^{n-2} w_{1i} w_{2i} \frac{c^i}{a^{n-1} - a^i}. \tag{3.9}$$

The quantity a_c plays a key role in our subsequent discussions. As mentioned in the beginning of this section, $N_{m,n,p}$ denotes the NNLM with m inputs, n hidden nodes, and p outputs.

4. CONTROLLABILITY AND OBSERVABILITY

The basic issues of controllability and observability for the system (3.8) will be discussed in this section. For discussion of controllability and observability, interested readers may refer to [46]. Basic definitions of controllability and observability are as follows: A system is said to be *controllable* at time t_0 if it is possible to transfer the system from any initial state $x(t_0)$ to any other state in a finite time interval via the use of an unconstrained control vector.

A system is said to be *observable* at time t_0 if, with the system in state $x(t_0)$, it is possible to determine this state from the observation of the output over a finite time interval. We provide the following theorem; for the proof of Theorem 1, also see [5,6].

THEOREM 1. *Suppose that*

(i) $w_{ij} \neq 0$ *for all* i, j,

(ii) $c^i \neq 0$ *for all* i,

(iii) $a_c \neq 0$ *and* $a_c < \infty$.

Then, the system (3.8) is completely controllable, if and only if, the following inequalities hold:

$$a^i \neq a^j \text{ for } i \neq j, \qquad i, j = 1, 2, \ldots, n-1. \tag{4.1}$$

PROOF. **Sufficiency.** We shall prove sufficiency by using the Popov-Belevitch-Hautus rank test (see [47]). Let A_1 be defined as follows:

$$A_1 = [sI - A \; B] = \begin{bmatrix} s-a^0 & 0 & 0 & \ldots & 0 & 0 & 1 \\ -w_{11}\,c^0\,a^0 & s-a^1 & 0 & \ldots & 0 & 0 & w_{11}\,c^0 \\ -w_{12}\,c^0\,a^0 & 0 & s-a^2 & \ldots & 0 & 0 & w_{12}\,c^0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -w_{1(n-2)}\,c^0\,a^0 & 0 & 0 & \ldots & s-a^{n-2} & 0 & w_{1(n-2)}\,c^0 \\ -a^0\tilde{a} & -w_{21}\,c^1\,a^1 & -w_{22}\,c^2\,a^2 & \ldots & -w_{2(n-2)}\,c^{n-2}\,a^{n-2} & s-a^{n-1} & \tilde{a} \end{bmatrix}.$$

Obviously, A_1 has rank n if s is not an eigenvalue of A_1. For $s = a^0$ and if $a^0 \neq a^i$ for $i \geq 1$, multiplying the last column by a^0 and adding it to the 1st column yields

$$A_2 = \begin{bmatrix} a^0 & 0 & 0 & \ldots & 0 & 1 \\ 0 & a^0-a^1 & 0 & \ldots & 0 & w_{11}\,c^0 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \ddots & 0 & w_{1(n-2)}\,c^0 \\ 0 & -w_{21}\,c^1\,a^1 & -w_{22}\,c^2\,a^2 & \ldots & a^0-a^{n-1} & \tilde{a} \end{bmatrix},$$

which has rank n.

For $s = a^0$ and if $a^0 = a^i$ for some i, deleting the n^{th} column of matrix A_1 yields a matrix A_3:

$$A_3 = \begin{bmatrix} 0 & 0 & 0 & \ldots & 0 & 1 \\ -w_{11}\,c^0\,a^0 & a^0-a^1 & 0 & \ldots & 0 & w_{11}\,c^0 \\ -w_{12}\,c^0\,a^0 & 0 & a^0-a^2 & \ldots & 0 & w_{12}\,c^0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -w_{1(n-2)}\,c^0\,a^0 & 0 & 0 & \ldots & a^0-a^{n-2} & w_{1(n-2)}\,c^0 \\ -a^0\tilde{a} & -w_{21}\,c^1\,a^1 & -w_{22}\,c^2\,a^2 & \ldots & -w_{2(n-2)}\,c^{n-2}\,a^{n-2} & \tilde{a} \end{bmatrix}.$$

After elementary transformations are performed on the matrix A_3, its last row becomes $[0, 0, \ldots, 0, *, 0, \ldots, 0]$. Then, performing another series of elementary transformations on the resulting matrix yields the following

$$A_4 = \begin{bmatrix} a^0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & a^0-a^1 & \ddots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & \cdots & a^0-a^{i-1} & \vdots & 0 & \cdots & 0 & * \\ 0 & \cdots & \cdots & 0 & \vdots & a^0-a^{i+1} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & a^0-a^{n-2} & 0 \\ 0 & 0 & \cdots & 0 & * & 0 & \cdots & 0 & 0 \end{bmatrix},$$

which has rank n.

For $s = a^i (1 \le i \le n-2)$, deleting the n^{th} column of A_1 yields

$$A_5 = \begin{bmatrix} a^i & 0 & 0 & \cdots & 0 \\ 0 & a^i - a^1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \vdots & a^i - a^{i-1} & 0 \\ \vdots & \vdots & \vdots & 0 & 0 \\ \vdots & \vdots & \vdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & -w_{21}\, c^1 a^1 & -w_{22}\, c^2 a^2 & \cdots & -w_{2(i-1)}\, c^{i-1} a^{i-1} \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 0 & w_{11} c^0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \vdots & \vdots & \vdots & w_{1i}\, c^0 \\ 0 & \ddots & \vdots & \vdots & w_{1(i+1)}\, c^0 \\ a^i - a^{i+1} & \vdots & \ddots & \vdots & w_{1(i+2)}\, c^0 \\ \vdots & \vdots & \vdots & a^i - a^{n-2} & \vdots \\ -w_{2i}\, c^i a^i & -w_{2(i+1)}\, c^{i+1} a^{i+1} & \cdots & -w_{2(n-2)}\, c^{n-2} a^{n-2} & \tilde{a} \end{bmatrix}.$$

After performing a series of elementary transformations, it is not hard to show that the rank of the matrix is again n.

For the case $s = a^{n-1}$, deleting the n^{th} column of A_1 and performing one elementary transformation on A_1 yields

$$A_6 = \begin{bmatrix} a^{n-1} & 0 & 0 & \cdots & 0 & 1 \\ 0 & a^{n-1} - a^1 & 0 & \cdots & 0 & w_{11} c^0 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \ddots & a^{n-1} - a^{n-2} & w_{1(n-2)}\, c^0 \\ 0 & -w_{21}\, c^1 a^1 & -w_{22}\, c^2 a^2 & \cdots & -w_{2(n-2)}\, c^{n-2} a^{n-2} & \tilde{a} \end{bmatrix}.$$

Again, performing another series of elementary transformations on A_6, one obtains

$$A_7 = \begin{bmatrix} a^{n-1} & 0 & 0 & \cdots & 0 & 1 \\ 0 & a^{n-1} - a^1 & 0 & \cdots & 0 & w_{11}\, c^0 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \ddots & a^{n-1} - a^{n-2} & w_{1(n-2)}\, c^0 \\ 0 & 0 & 0 & \cdots & 0 & \tilde{a_c} \end{bmatrix},$$

where $\tilde{a_c}$ is

$$\tilde{a_c} = a_c\, c^0\, a^{n-1}. \tag{4.2}$$

Therefore, A_7 has also rank n.

Necessity. Necessity is proved by contradiction. Letting $a^i = a^j$ for some $i \neq j$, $i, j = 1, 2, \ldots, n-1$ yields a matrix $A^{\#}$ which has rank less than n. ∎

REMARKS. It is easy to see from the proof that the condition $a^0 = a^i (1 \le i \le n-1)$ is allowed. Thus, the system is still controllable even for repeated eigenvalues $a^0 = a^i$ for some i between 1 and $n-1$. Notice that $a_{n-1} \neq a_i$ for $i = 0, \ldots, n-2$ is only a sufficient condition for the theorem. The following example shows that the assumption may not be necessary.

Considering a case where $n = 2$, we have the state equations

$$x_k^0 = a^0 x_{k-1}^0 + u_k, \tag{4.3}$$

$$x_k^1 = a^1 x_{k-1}^1 + w_{11} c^0 x_{k-1}^0, \tag{4.4}$$

$$y_k^1 = c^1 x_k^1. \tag{4.5}$$

This system is controllable regardless of what the values of a^0 and a^1 are. Thus, letting $a^0 = a^1 = \text{constant}$, we still have a controllable system. A similar result regarding the observability of the system is obtained.

THEOREM 2. *Suppose that*

(i) $w_{ij} \neq 0$, *for all* i, j,
(ii) $c^i \neq 0$, *for all* i,
(iii) $a_c \neq 0$ *and* $a_c < \infty$.

Then, system (3.8) is completely observable if and only if the following inequalities hold

$$a^i \neq a^j, \qquad \text{for } i \neq j, \quad i, j = 1, 2, \ldots, n-1. \tag{4.6}$$

PROOF. Necessity and sufficiency can be proved again by using the Popov-Belevitch-Hautus rank test, namely, by checking the rank of the matrix $[C^\top \ (sI - A)^\top]^\top$. Similar arguments lead to the conclusion of this theorem, with the only difference being that the column transformations are changed to corresponding row transformations. ∎

REMARK. The result on observability of the systems holds only under the assumption that the activation functions of all node systems are linear.

Based on the analysis on controllability and observability, we have the following.

THEOREM 3. *Suppose that*

(i) *all activation functions* s_j *are linear*,
(ii) $w_{ij} \neq 0$, *for all* i, j,
(iii) $c^i \neq 0$, *for all* i,
(iv) $a_c \neq 0$ *and* $a_c < \infty$,
(v) $a^i \neq a^j$, *for* $i \neq j$.

Then, any strictly proper SISO linear system with real and nonrepeating eigenvalues can be realized by an $N_{1,n-2,1}$, *where the* a^i*'s are the eigenvalues of the system.*

PROOF. Because system (3.8) is completely controllable and observable, the transfer function $\mathbf{C}(sI - \mathbf{A})^{-1}\mathbf{B}$ has no pole-zero cancellation between its numerator and denominator. Since the order of the denominator polynomial is n, it represents a typical n^{th}-order rational transfer function. ∎

We give an example for the case $n = 4$. The matrices $\mathbf{A}$ and $\mathbf{B}$ in this case are

$$\mathbf{A} = \begin{bmatrix} a^0 & 0 & 0 & 0 \\ w_{11}c^0a^0 & a^1 & 0 & 0 \\ w_{12}c^0a^0 & 0 & a^2 & 0 \\ a^0\tilde{a} & w_{21}c^1a^1 & w_{22}c^2a^2 & a^3 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} 1 \\ w_{11}c^0 \\ w_{12}c^0 \\ \tilde{a} \end{bmatrix}, \qquad \mathbf{C} = [0 \quad 0 \quad 0 \quad c^3],$$

where $\tilde{a} = w_{21}\, c^1\, w_{11}\, c^0 + w_{22}\, c^2\, w_{12}\, c^0$ and the transfer function is

$$\frac{c^3 \left(b_3\, s^3 + b_2\, s^2 + b_1\, s + b_0\right)}{(s - a^0)\,(s - a^1)\,(s - a^2)\,(s - a^3)}, \tag{4.7}$$

and

$$
\begin{aligned}
b_3 &= \tilde{a}, \\
b_2 &= -\tilde{a}(a^0 + a^1 + a^2) + a^2 c^0 c^2 w_{12} w_{22} + a^1 c^0 c^1 w_{11} w_{21} + a^0 \tilde{a}, \\
b_1 &= \tilde{a}(a^0 a^2 + a^0 a^1 + a^1 a^2) - (a^0 + a^1) a^2 c^0 c^2 w_{12} w_{22} - (a^0 + a^2) a^1 c^0 c^1 w_{11} w_{21} \\
&\quad - a^0 a^1 c^0 c^2 w_{12} w_{22} - a^0 a^2 \tilde{a}, \\
b_0 &= -\tilde{a} a^0 a^1 a^2 + a^0 a^1 a^2 c^0 c^2 w_{12} w_{22} + a^0 a^1 a^2 c^0 c^1 w_{11} w_{21} + a^0 a^1 a^2 c^0 c^2 w_{12} w_{22}.
\end{aligned}
$$

Thus, by properly choosing $w_{11}, w_{12}, w_{21}, w_{22}$, we can realize a fourth-order linear system. A block diagram for the realization is shown in Figure 4.

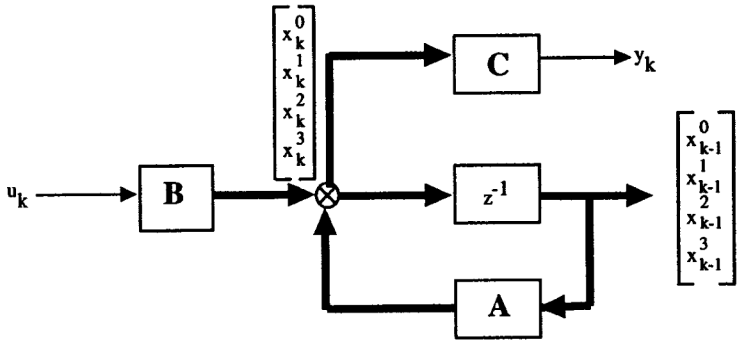


Figure 4. Linear system representation of order four.

In what follows, we shall discuss the effects of the weights on the overall performance of the system. So far we have shown that some of the entries of matrices **A** and **B** in (3.8) contain the weights of the network. This seems to imply that the weights could affect the dynamics of the system. However, this turns out not to be the case. In fact, the transfer function (4.7) tells us that the weights of the network will affect only the numerator of the system and do not affect the eigenvalues of the system. In general, we have the following

$$\text{Transfer Function} = \frac{d(s; \mathbf{w}, \mathbf{a}, \mathbf{c})}{\prod_{i=0}^{n-1}(s - a_i)}, \tag{4.8}$$

where $\mathbf{w} = (w_{ij})_{n \times n}$, $\mathbf{a} = (a^0, \ldots, a^{n-1})$, $\mathbf{c} = (c^0, c^1, \ldots, c^{n-1})$ and $d(s; \mathbf{w}, \mathbf{a}, \mathbf{c})$ is a polynomial of order $n-1$ whose coefficients are the linear combination of entries of matrices $\mathbf{w}$, $\mathbf{a}$, and $\mathbf{c}$. This property will be formally stated as follows.

PROPERTY 1. *The dynamics of the system will not be affected by changing the weights of the network.*

Based on this property and the fact that the NLMs are extensions of the McCulloch-Pitts neurons, we obtain the Separation Principle of Learning and Control, stated below. The importance of this principle lies in the fact that before we actually use the system, we can set all a^i's to be zero. We then train the network using the back-propagation algorithm with a prespecified training set so that the network has the desired stationary property. After training is done, the parameters a^i can be resumed and the network will function as a normal system.

Separation Principle of Learning and Control: The Training Process of an NNLM and the Control Process after Training Can Be Separated

To illustrate this idea, let us now consider a small network with one hidden layer and two hidden neurons (see Figure 5). Using the same notations, we may have the following set of

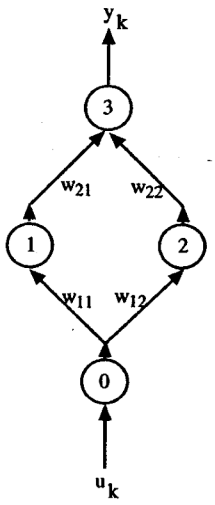


Figure 5. A simple network.

equations for each neuron:

$$\text{node } 0: \begin{cases} x_k^0 &= a^0 x_{k-1}^0 + u_k, \\ y_k^0 &= s_0\left(c^0 x_k^0\right), \end{cases}$$

$$\text{node 1 and node 2}: \begin{cases} x_k^i &= a^i x_{k-1}^i + w_{1i} y_k^0, \\ y_k^i &= s_2\left(c^i x_k^i\right), \qquad i = 1, 2, \end{cases}$$

$$\text{node } 3: \begin{cases} x_k^3 &= a^3 x_{k-1}^3 + w_{21} y_k^1 + w_{22} y_k^2, \\ y_k &= s_3\left(c^{n-1} x_k^{n-1}\right), \end{cases}$$

where superscript i is the index for the i^{th} neuron. With this network, let us now consider a simple tracking problem. The input to the plant is given by $u_k = \bar{u}$, for all $k \geq 0$. Our goal is to keep the difference between the actual output and the desired signal y_d as small as possible, where y_d is a constant. Usually, we need to analyze the steady state of the output of the system. In our case, we shall perform the same analysis and assume that all state variables have reached their steady states. Denote x^i as the steady state for the i^{th} state variable, and $\bar{y}$ as the steady state for the output y_k. Again, for simplicity, we assume that the activation function $s_0(.)$ for input node is linear. With these notations, we have the following:

$$\begin{aligned} x^0 &= \frac{\bar{u}}{1-a^0}, \\ x^1 &= \frac{w_{11}\, c^0 \bar{u}}{(1-a^1)(1-a^0)}, \\ x^2 &= \frac{w_{12}\, c^0 \bar{u}}{(1-a^2)(1-a^0)}, \\ x^3 &= \frac{w_{21}}{1-a^3} s_2\left(\frac{w_{11}\, c^0 c^1 \bar{u}}{(1-a^0)(1-a^1)}\right) + \frac{w_{22}}{1-a^3} s_2\left(\frac{w_{12}\, c^0 c^2 \bar{u}}{(1-a^0)(1-a^2)}\right), \\ \bar{y} &= s_3\left(c^3 x^3\right) \\ &= s_3\left[\frac{c^3 w_{21}}{1-a^3} s_2\left(\frac{w_{11}\, c^0 c^1\, \bar{u}}{(1-a^0)(1-a^1)}\right) + \frac{c^3 w_{22}}{1-a^3} s_2\left(\frac{w_{12}\, c^0 c^2\, \bar{u}}{(1-a^0)(1-a^2)}\right)\right]. \end{aligned} \tag{4.9}$$

(4.9)(cont.)

Now, let us define the following weights w_1', w_2', w_3', w_4' as

$$w_1' = \frac{w_{11}\, c^1}{1-a^1}, \qquad w_2' = \frac{w_{12}\, c^2}{1-a^2}, \qquad w_3' = \frac{w_{21}\, c^3}{1-a^3}, \qquad w_4' = \frac{w_{22}\, c^3}{1-a^3}, \qquad \bar{u}' = \frac{c^0 \bar{u}}{1-a^0}.$$

Then, equation (4.9) becomes

$$\bar{y} = s_3 \left[w_3' \, s_2 \left(w_1' \bar{u}' \right) + w_4' \, s_2 \left(w_2' \bar{u}' \right) \right]. \tag{4.10}$$

Obviously, equation (4.10) is the conventional expression of this network, when $a^i = 0, c^i = 1$ for $i = 1, 2, 3$, with which we are very familiar. This is exactly the input-output relationship of a typical feed-forward neural network. Thus, we may construct a training set consisting of the input line being the value of $\bar{u}'$, and the desired output line being y_d. Then we can use a learning algorithm, for example the back-propagation learning algorithm, to train the network such that the output of the trained network is as close as to y_d as possible. Once we have trained the network, we know that the system—the network with a^i and c^i being resumed—has the desired stationary property when the state variables reach their steady states as long as we set w_1, w_2, w_3, and w_4 as

$$w_{11} = \frac{1}{c^1} \left(1 - a^1 \right) w_1', \quad w_{12} = \frac{1}{c^2} \left(1 - a^2 \right) w_2',$$
$$w_{21} = \frac{1}{c^3} \left(1 - a^3 \right) w_3', \quad w_{22} = \frac{1}{c^3} \left(1 - a^2 \right) w_4'.$$

where w_1', w_2', w_3', w_4' are the weights after the network has been trained. After the training, a^i and c^i can be resumed to their normal values and the network is ready to be used as a control system.

5. LINEARIZATION VIA TRANSFORMATION OF COORDINATES AND NONLINEAR FEEDBACK

In Section 4, we saw that our representation resulted in a nonlinear discrete-time system. There are many reasons for linearizing a nonlinear system, and many publications in the literature [25,26,46,48,49] discuss this problem. Before we proceed, let us look at our discrete-time system whose nonlinearity arises from the nonlinear activation functions. In general, the activation functions in input nodes of a neural network are linear. Thus, the state-space description of our system has the form

$$\begin{aligned}
x_k^0 &= a^0 \, x_{k-1}^0 + u_k, \\
x_k^1 &= w_{11} \, c^0 \, a^0 \, x_{k-1}^0 + a^1 \, x_{k-1}^1 + w_{11} \, c^0 \, u_k, \\
&\vdots \\
x_k^{n-2} &= w_{1(n-2)} \, c^0 \, a^0 \, x_{k-1}^0 + a^{n-2} \, x_{k-1}^{n-2} + w_{1(n-2)} \, c^0 \, u_k, \\
x_k^{n-1} &= a^{n-1} \, x_{k-1}^{n-1} + \sum_{i=1}^{n-2} w_{2i} \, s_2 \left(c^i \, a^i \, x_{k-1}^i + c^i \, w_{1i} \, c^0 \, a^0 \, x_{k-1}^0 + c^i \, w_{1i} \, c^0 \, u_k \right), \\
y_k &= s_3 (c^{n-1} x_k^{n-1}).
\end{aligned} \tag{5.1}$$

The above equations can be written in the following form:

$$\mathbf{x}_k = \mathbf{f}(x_{k-1}, u_k), \tag{5.2}$$

where $\mathbf{x}_k = (x_k^0, \ldots, x_k^{n-1})$ and $\mathbf{f}(x_{k-1}, u_k) = (f_0(\mathbf{x}_{k-1}, u_k), \ldots, f_{n-1}(\mathbf{x}_{k-1}, u_k))$ is a vector of the functions which are defined above.

From the above, we know that the overall system consists of a linear subsystem cascaded by a nonlinear subsystem together with a nonlinear output equation (see Figure 6). This, in turn, implies that the overall system is a nonlinear one.

It is natural to consider the problem of locally linearizing the above system via coordinate transformations and nonlinear feedback. In general, not all nonlinear systems can be so linearized.

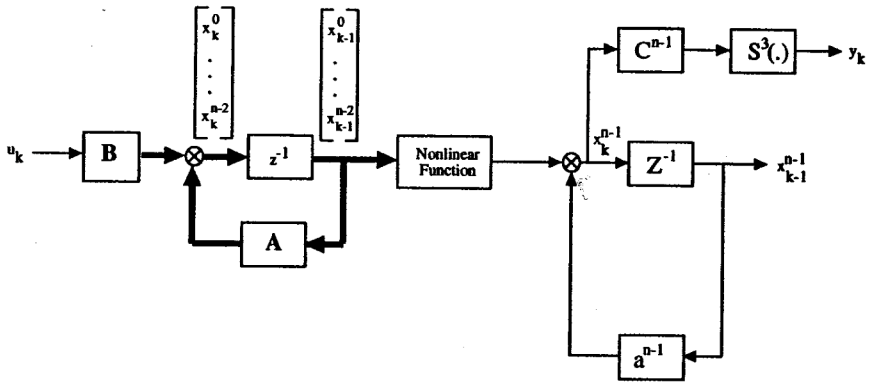


Figure 6. Nonlinear system representation of order n.

A necessary and sufficient condition will be given in Section 5.2. Once a linearized system is obtained, it is not too difficult to implement a nonlinear control law to have the system track some desired signal.

5.1. Preliminary

We consider a smooth discrete-time nonlinear dynamic system

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_k), \tag{5.3}$$

where $\mathbf{x}_k = (x_k^0, x_k^1, \ldots, x_k^{n-1})$ and $\mathbf{u}_k = (u_k^0, u_k^1, \ldots, u_k^{m-1})$ are smooth local coordinates for the state-space M and input space U, respectively. Before discussing feedback linearizability for (5.3), we introduce the notion of a regular static state feedback. We call a relation

$$\mathbf{u}_k = \alpha(\mathbf{x}_{k-1}, \mathbf{v}_k), \tag{5.4}$$

a regular static state feedback whenever $\frac{\partial \alpha}{\partial v}(\mathbf{x}_{k-1}, \mathbf{v}_k)$ is nonsingular at every point $(\mathbf{x}_{k-1}, \mathbf{v}_k)$. Notice that this implies locally a one-to-one relation between the old inputs $\mathbf{u}_k$ and the new controls $\mathbf{v}_k$. We can now formulate the notion of feedback linearizability for (5.3).

DEFINITION 1. *Let $(\mathbf{x}_0, \mathbf{u}_0)$ be an equilibrium point for (5.3), i.e., $\mathbf{x}_0 = f(\mathbf{x}_0, \mathbf{u}_0)$. The system (5.3) is feedback linearizable around $(\mathbf{x}_0, \mathbf{u}_0)$ if there exists*

(a) *a coordinate transformation $S : V \in R^n \to S(V) \subset R^n$ defined on a neighborhood V of $\mathbf{x}_0$ with $S(\mathbf{x}_0) = \mathbf{0}$;*

(b) *a regular feedback $\mathbf{u} = \alpha(\mathbf{x}, \mathbf{v})$ satisfying $\alpha(\mathbf{x}_0, \mathbf{0}) = \mathbf{u}_0$ and defined on a neighborhood $V \times O$ of $(\mathbf{x}_0, \mathbf{0})$ with $\frac{\partial \alpha}{\partial v}(\mathbf{x}, \mathbf{v})$ nonsingular on $V \times O$,*

such that in the new coordinates $\mathbf{z} = S(\mathbf{x})$, the closed loop dynamics are linear

$$\mathbf{z}_k = A\mathbf{z}_{k-1} + B\mathbf{v}_k, \tag{5.5}$$

for some matrices A and B.

At this point, let us look at the equilibrium points of our nonlinear system. For the system (5.3), it is not difficult to show that the $\mathbf{x}^*, \mathbf{u}^*$ satisfying $f(\mathbf{x}^*, \mathbf{u}^*) = \mathbf{x}^*$ have the form

$$\begin{aligned} x^{0*} &= \frac{u^*}{1 - a^0}, \\ x^{1*} &= \frac{1}{1 - a^1}\left[w_{11}\, c^0\, a^0\, x^{0*} + w_{11}\, c^0\, u^*\right], \\ &\vdots \\ x^{(n-1)*} &= \frac{1}{1 - a^{n-1}}\left[\sum_{i=1}^{n-2} w_{2i}\, s_2\left(c^i\, a^i\, x^{i*} + c^i\, w_{1i}\, c^0\, a^0\, x^{0*} + c^i\, w_{1i}\, c^0\, u^*\right)\right]. \end{aligned} \tag{5.6}$$

Therefore, $x^{0*}, x^{1*}, \ldots, x^{(n-2)*}$ are all linear functions of u^* but $x^{(n-1)*}$ is not.

5.2. Necessary and Sufficient Conditions for Local Linearization via Transformation of Coordinates and Nonlinear Feedback

In this section, we will use Grizzle's necessary and sufficient conditions [50] to prove that our nonlinear system is locally linearizable to a controllable linear system.

Before we formally give the result in the next section, let us look at a sequence of distributions given by Grizzle in [50]. This sequence will be instrumental in the solution of the feedback linearization problem for (5.3).

Let $\pi : M \times U \to M$ be the canonical projection and K the distribution defined by

$$K = \ker f_*, \tag{5.7}$$

where $M \subset R^n$, $U \subset R^m$ and f_* is the dual vector space homomorphism from $TM \times TU$ to TM.

ALGORITHM 5.1. Assume f_* has full rank around $(\mathbf{x}_0, \mathbf{u}_0)$.

STEP 0. Define the distribution D_0 in a neighborhood of $(\mathbf{x}_0, \mathbf{u}_0)$ in $M \times U$ by

$$D_0 = \pi_*^{-1}(0). \tag{5.8}$$

STEP I+1. Suppose that around $(\mathbf{x}_0, \mathbf{u}_0)$ $D_i + K$ is an involutive constant-dimensional distribution on $T(M \times U)$. Then define in a neighborhood of $(\mathbf{x}_0, \mathbf{u}_0)$

$$D_{i+1} = \pi_*^{-1} f_*(D_i), \tag{5.9}$$

and stop if $D_i + K$ is not involutive or constant-dimensional.

The effectiveness of the above algorithm rests upon the following observation.

LEMMA 1. *Let (x_0, u_0) be an equilibrium point of (5.3), and assume that f_* has full rank around (x_0, u_0). Let D be an involutive constant-dimensional distribution on $M \times U$ such that $D + K$ is also involutive and constant-dimensional. Then there exists a neighborhood O of (x_0, u_0) such that $f_*(D|_O)$ is an involutive constant-dimensional distribution around x_0.*

Based on the above algorithm and lemma, Grizzle [50] states necessary and sufficient conditions for locally linearizing a nonlinear system to a controllable one.

THEOREM 4. *(See [50].) Consider the discrete-time nonlinear system (5.3), about the equilibrium point (x_0, u_0). The system (5.3) is linearizable around (x_0, u_0) to a controllable linear system if and only if Algorithm 5.1 applied to the system (5.3) gives distributions $D_0, \ldots, D_n$ such that* $\dim(D_n) = n + m$.

The proof of the above lemma and theorem can be found in [50]. In the next section, we are going to show that our nonlinear system satisfies the conditions of the above theorem and thus, the system is locally linearizable.

5.3. Main Result

Now, let us consider our nonlinear system (5.3) in which $f(x, u)$ has the form

$$f(x,u) = \begin{bmatrix} a^0\, x^0 + u \\ w_{11}\, c^0\, a^0\, x^0 + a^1\, x^1 + w_{11}\, c^0\, u \\ \vdots \\ w_{1(n-2)}\, c^0\, a^0\, x^0 + a^{n-2}\, x^{n-2} + w_{1(n-2)}\, c^0\, u \\ a^{n-1}\, x^{n-1} + \sum_{i=1}^{n-2} w_{2i}\, s_2 \left(c^i\, a^i\, x^i + c^i\, w_{2i}\, c^0\, a^0\, x^0 + c^i\, w_{1i}\, c^0\, u \right) \end{bmatrix}, \tag{5.10}$$

where $x = (x^0, x^1, \ldots, x^{n-1})$ and u is a scalar. Before we present the main theorem, we shall state and prove some lemmas, which will be used later.

LEMMA 2. *Consider the nonlinear system (5.3) and the nonlinear function $f(x,u)$ of (5.10). If $a^i \neq 0$ for $0 \leq i \leq n-1$, then f_* has full rank around the equilibrium point (x^*,u^*).*

PROOF. By noting that $f : M \times U \to M$ is given by (5.10), we can evaluate $f_* : TM \times TU \to TM$ by considering the natural basis $(\frac{\partial}{\partial x^0},\ldots,\frac{\partial}{\partial x^{n-1}})$ in TM and $\frac{\partial}{\partial u}$ in TU, where $M \in R^n$, $U \in R$, and TM and TU are the tangent spaces for M and U, respectively. Let $Z^1, Z^2, \ldots, Z^n$ be the basis in the image of f_*. Then,

$$f_* \begin{pmatrix} \dfrac{\partial}{\partial x^0} \\ \dfrac{\partial}{\partial x^1} \\ \vdots \\ \dfrac{\partial}{\partial u} \end{pmatrix} = A \begin{pmatrix} Z_1 \\ Z_2 \\ \vdots \\ Z_n \end{pmatrix},$$

where $A = (a_{ij})$, given by $a_{ij} = \frac{\partial f^j}{\partial x^i}$, $i,j = 0,1,\ldots,n-1$, and $a_{nk} = \frac{\partial f^k}{\partial u}$, $k = 0,1,\ldots,n-1$. Thus,

$$A = \begin{bmatrix} a^0 & w_{11}\, c^0\, a^0 & \cdots & w_{1(n-2)}\, c^0\, a^0 & \sum_{i=1}^{n-2} w_{2i}\, s_2'(.)\, c^i\, w_{1i}\, c^0\, a^0 \\ 0 & a^1 & \cdots & 0 & w_{21}\, s_2'(.)\, c^1\, a^1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & a^{n-2} & w_{2(n-2)}\, s_2'(.)\, c^{n-2}\, a^{n-2} \\ 0 & 0 & \cdots & 0 & a^{n-1} \\ 1 & w_{11}\, c^0 & \cdots & w_{1(n-2)}\, c^0 & \sum_{i=1}^{n-2} w_{2i}\, s_2'(.)\, c^i\, w_{1i}\, c^0 \end{bmatrix}. \tag{5.11}$$

Since $a^i \neq 0$ for $0 \leq i \leq n-1$, $\operatorname{rank}(A) = n$ and thus f_* has full rank around (x^*,u^*). ∎

REMARK. In the subsequent analysis, we shall see that the matrix A plays an important role, especially in the case where an NNLM with more than one hidden layer is considered.

LEMMA 3. *Let the conditions in Lemma 2 be satisfied. Let D be a subspace in $TM \times TU$. Then*

$$\dim(f_*(D)) = \begin{cases} \dim(D), & \text{if } \dim(D) \leq n, \\ n, & \text{if } \dim(D) = n+1. \end{cases}$$

PROOF.

CASE (i). $\dim(D) \leq n$: Suppose that $\dim(D) = p \leq n$ and let $Y^1, Y^2, \ldots, Y^p$ be the basis in D. Then, without loss of generality, we have

$$\begin{pmatrix} Y^1 \\ Y^2 \\ \vdots \\ Y^p \end{pmatrix} = P \begin{pmatrix} \dfrac{\partial}{\partial x^0} \\ \dfrac{\partial}{\partial x^1} \\ \vdots \\ \dfrac{\partial}{\partial x^{n-1}} \\ \dfrac{\partial}{\partial u} \end{pmatrix}, \tag{5.12}$$

where $P \in R^{p \times (n+1)}$ and $\operatorname{rank}(P) = p$. Then,

$$f_*\begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_p \end{pmatrix} = Pf_*\begin{pmatrix} \dfrac{\partial}{\partial x^0} \\ \dfrac{\partial}{\partial x^1} \\ \vdots \\ \dfrac{\partial}{\partial x^{n-1}} \\ \dfrac{\partial}{\partial u} \end{pmatrix} = PA\begin{pmatrix} Z_1 \\ Z_2 \\ \vdots \\ Z_n \end{pmatrix} = \hat{P}\begin{pmatrix} Z_1 \\ Z_2 \\ \vdots \\ Z_n \end{pmatrix} = \begin{pmatrix} W_1 \\ W_2 \\ \vdots \\ W_p \end{pmatrix}.$$

To prove that $f_*(D) = \operatorname{span}(W_1, \ldots, W_p)$, it suffices to show that $\operatorname{rank}(\hat{P}) = p$. Indeed, the fact that $\operatorname{rank}(P) = \mathrm{p} \leq n$ and $\operatorname{rank}(A) = n$ implies that $\operatorname{rank}(\hat{P}) = p$. Thus, $\dim(f_*(D)) = \dim(D)$.

CASE (ii). $\dim(D) = n + 1$: Then (5.12) still holds, but in this case, $P \in R^{(n+1)\times(n+1)}$ and $\operatorname{rank}(P) = n + 1$. The fact that $\operatorname{rank}(\hat{P})$ follows from Sylvester's inequality on the rank of the product of two matrices and the rank inequality for matrices $A \in R^{n\times n}, B \in R^{n\times m}(m \leq n)$

$$\operatorname{rank}(A) + \operatorname{rank}(B) - n \leq \operatorname{rank}(AB).$$

Therefore, $\dim(f_*(D)) = n$. ∎

LEMMA 4. *Let π denote the canonical projection from $M \times U$ onto M given by $\pi(x, u) = x$, and let Q be a subset of TM with $\dim(Q) = p \leq n$. Then $\dim(\pi_*^{-1}(Q)) = p + 1$.*

PROOF. Let $Y_1, \ldots, Y_p$ be a basis in Q. Then any vector field in Q can be represented by $\sum_{i=1}^{p} a_i Y_i$, that is,

$$(a^1, \ldots, a^p, 0, \ldots, 0)\begin{pmatrix} Y_1 \\ \vdots \\ Y_p \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

But

$$\begin{pmatrix} Y_1 \\ \vdots \\ Y_p \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} P_1 & P_2 \\ 0 & P_3 \end{pmatrix}\begin{pmatrix} \dfrac{\partial}{\partial x^1} \\ \vdots \\ \dfrac{\partial}{\partial x^n} \\ 0 \end{pmatrix} = P\begin{pmatrix} \dfrac{\partial}{\partial x^1} \\ \vdots \\ \dfrac{\partial}{\partial x^n} \\ 0 \end{pmatrix},$$

where $P_1 \in R^{p\times n}$, $P_2 \in R^{p\times 1}$, $P_3 \in R^{(n+1-p)\times 1}$ and $\operatorname{rank}(P_1) = p$, $\operatorname{rank}(P_2) = 1$.

Let $[\frac{\partial}{\partial x^1}, \ldots, \frac{\partial}{\partial x^n}, \frac{\partial}{\partial u}]^{\mathrm{T}}$ be a basis in $TM \times TU$, then

$$\pi_*\begin{pmatrix} \dfrac{\partial}{\partial x^1} \\ \vdots \\ \dfrac{\partial}{\partial x^n} \\ \dfrac{\partial}{\partial u} \end{pmatrix} = I\begin{pmatrix} \dfrac{\partial}{\partial x^1} \\ \vdots \\ \dfrac{\partial}{\partial x^n} \\ 0 \end{pmatrix},$$

where I is an identity matrix. Thus,

$$\pi_*^{-1}\begin{pmatrix}\hat{a}_1 & \cdots & \hat{a}_n & a\end{pmatrix}\begin{pmatrix}\frac{\partial}{\partial x^0}\\ \frac{\partial}{\partial x^1}\\ \vdots\\ \frac{\partial}{\partial x^{n-1}}\\ 0\end{pmatrix} = \begin{pmatrix}\hat{a}_1 & \cdots & \hat{a}_n & a\end{pmatrix}\pi_*^{-1}\begin{pmatrix}\frac{\partial}{\partial x^1}\\ \vdots\\ \frac{\partial}{\partial x^n}\\ 0\end{pmatrix}$$

$$= \begin{pmatrix}\hat{a}_1 & \cdots & \hat{a}_n & a\end{pmatrix}\begin{pmatrix}\frac{\partial}{\partial x^1}\\ \vdots\\ \frac{\partial}{\partial x^n}\\ \frac{\partial}{\partial u}\end{pmatrix} = \sum_{i=1}^{n}\hat{a}_i\frac{\partial}{\partial x^i} + a\frac{\partial}{\partial u},$$

and

$$\pi_*^{-1}\begin{pmatrix}Y_1\\ \vdots\\ Y_p\\ 0\\ \vdots\\ 0\end{pmatrix} = P\pi_*^{-1}\begin{pmatrix}\frac{\partial}{\partial x^1}\\ \vdots\\ \frac{\partial}{\partial x^n}\\ 0\end{pmatrix} = PI\begin{pmatrix}\frac{\partial}{\partial x^1}\\ \vdots\\ \frac{\partial}{\partial x^n}\\ \frac{\partial}{\partial u}\end{pmatrix}.$$

So $\operatorname{rank}(P) = p+1$ implies that

$$\pi_*^{-1}\begin{pmatrix}Y_1\\ \vdots\\ Y_p\\ 0\\ \vdots\\ 0\end{pmatrix}$$

has dimension $p+1$ or $\dim(\pi_*^{-1}(Q)) = p+1$. ∎

THEOREM 5. *Consider the discrete-time nonlinear system (5.3) about the equilibrium point* (x^*, u^*). *If* $a^i \neq 0$ *for* $0 \leq i \leq n-1$, *then the system is linearizable around* (x^*, u^*) *to a controllable linear system.*

PROOF.

STEP 1. Using Lemma 2, we see that f_* has full rank around the equilibrium (x^*, u^*). Therefore, we can apply Algorithm 5.1 to compute D_i.

Let $K = \ker f_*$. Note that $f_* : TM \times TU \to TM$, and $TM \times TU \subset R^n \times R$ and $TM \subset R^n$. Therefore, $f_*(\bar{a}^1, \ldots, \bar{a}^n, \bar{a}) = (\bar{a}^1, \ldots, \bar{a}^n, \bar{a})A$. The equality $f_*(\bar{a}^1, \bar{a}^2, \ldots, \bar{a}^n, \bar{a}) = \mathbf{0}$ implies that

$$(\bar{a}^1, \ldots, \bar{a}^n, \bar{a})\, A = \mathbf{0}. \tag{5.13}$$

Let

$$T = \begin{bmatrix} 1 & -w_{11}\, c^0 & \cdots & -w_{1(n-2)}\, c^0 & -\sum_{i=1}^{n-2} w_{1i}\, s_2(.) c^i\, w_{1i}\, c^0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \tag{5.14}$$

and

$$A^* = AT = \begin{bmatrix} a^0 & 0 & \cdots & 0 & 0 \\ 0 & a^1 & \cdots & 0 & w_{21}s_2'(.)c^1a^1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & a^{n-1} \\ 1 & 0 & \cdots & 0 & 0 \end{bmatrix}.$$

Now right-multiplying (38) by T yields:

$$(\bar{a}^1 \quad \cdots \quad \bar{a}^n \quad \bar{a})\, A^* = 0,$$

or

$$(\bar{a} + a^0\bar{a}^1 \quad a^1\bar{a}^2 \quad \cdots \quad a^{n-2}\bar{a}^{n-1} \quad \textstyle\sum_{i=1}^{n-2} w_{2i}\, s_2'(.)\, c^i\, a^i\bar{a}^{i+1} + a^{n-1}\bar{a}^n) = 0,$$

from which we conclude that $\bar{a}^2 = \cdots = \bar{a}^{n-1} = \bar{a}^n = 0$. But $\bar{a} = -a^0\bar{a}^1$. So,

$$\begin{aligned} K &= \text{span}\left(\bar{a}^1\frac{\partial}{\partial x^1} - a^0\bar{a}^1\frac{\partial}{\partial u}\right), \\ &= \text{span}\left(\tilde{Y}\right), \end{aligned}$$

where $\tilde{Y} = \frac{\partial}{\partial x^1} - a^0\frac{\partial}{\partial u}$ and $\dim(K) = 1$.

STEP 2. Let $D_0 = \pi_*^{-1}(0)$. Then, we have $D_0 = \text{span}(\frac{\partial}{\partial u})$ and $\dim(D_0)$=1 from Lemma 4. Let $D_{i+1} = \pi_*^{-1} f_*(D_i)$ for $0 \le i < n$.

Suppose now $\dim(D_i) = p$ and $X_1, \ldots, X_p$ are a basis for D_i. Then we have

$$\begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_p \end{bmatrix} = P \begin{bmatrix} \frac{\partial}{\partial x^1} \\ \vdots \\ \frac{\partial}{\partial x^n} \\ \frac{\partial}{\partial u} \end{bmatrix},$$

where $P \in R^{p\times(n+1)}$. Thus, $[X_1, \ldots, X_p, \tilde{Y}]$ is a basis for $D_i + K$, and

$$\begin{bmatrix} X_1 \\ \vdots \\ X_p \\ \tilde{Y} \end{bmatrix} = \begin{bmatrix} P \\ \cdots \\ \bar{p} \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x^1} \\ \vdots \\ \frac{\partial}{\partial x^n} \\ \frac{\partial}{\partial u} \end{bmatrix},$$

where $\bar{p} = [1\ 0\ 0\ \cdots\ 0\ -a^0]$. Obviously, $[X_i, X_j] = 0$ for $i \neq j$ and $[X_i, \tilde{Y}] = 0$ for all i. Therefore, $D_i + K$ is involutive and has constant dimension. Repeatedly applying Lemma 2 and Lemma 3 on D_i, and using induction on i, we obtain a D_n whose dimension is $n+1$. It follows from Grizzle's necessary and sufficient condition that the nonlinear system is linearizable to a controllable linear system. ∎

REMARK. In the proofs of the above lemmas and the main theorem, no restrictions were placed on the activation functions. Thus, we may employ any type of continuously differentiable functions, though typically these functions are sigmoidal.

5.4. A More General Case

In a more general case, we consider an NNLM with one input, L hidden layers and one output layer. In the j^{th} hidden layer, where $j = 1, 2, \ldots, L$, denote the number of neurons as l_j. Then, the total number of hidden neurons is $n = l_1 + l_2 + \cdots + l_L$. The total number of equations for our system becomes $n + 2$.

To be more specific, we have the input node equation

$$x_k^0 = a^0 x_{k-1}^0 + u_k, \tag{5.15}$$

and the output node equation

$$x_k^{n+2} = a^{n+2} x_{k-1}^{n+2} + \sum_{j=1}^{l_L} w_{Lj,n+2} y_k^{L_j}, \tag{5.16}$$

where $y_k^{L_j}$'s are the outputs of the neurons in the L^{th} hidden layer and $w_{Lj,n+2}$'s are the weights connecting the L^{th} hidden layer to the output layer.

The hidden node equations are more complicated. In the j^{th} hidden layer, the i^{th} node equation is represented by

$$x_k^{ji} = a^{ji} x_{k-1}^{ji} + \sum_{q=1}^{l_j - 1} w_{(j-1)q,i} y_k^{(j-1)q}, \tag{5.17}$$

where $i = 1, \ldots, l_j$; $j = 1, \ldots, L$, and

$$y_k^{(j-1)q} = s_{j-1}\left(c^{(j-1)\,q} x_k^{(j-1)\,q}\right), \tag{5.18}$$

where s_{j-1} is the activation function in the $(j-1)^{\text{th}}$ layer. $c^{(j-1)\,q}$ is the scalar for the q^{th} neuron in the $(j-1)^{\text{th}}$ layer.

With these equations, we can look at the matrix A defined in equation (5.11). In this case, the matrix A is an $(n+3) \times (n+2)$ matrix, that is $A = (a_{ij})_{(n+3)\times(n+2)}$. If we define a submatrix of A as $A_1 = (\hat{a}_{ij})_{(n+2)\times(n+2)}, \hat{a}_{ij} = a_{ij}$, $i, j = 1, 2, \ldots, n+2$, then A_1 is a triangle matrix with diagonal elements being $a^0, \ldots, a^{n+2}$. By carefully examining the proofs of the Lemmas 2–4, we can conclude that Lemmas 2–4 still hold in this more general case. However, the proof of Theorem 1 becomes much more complicated.

6. ILLUSTRATION AND SIMULATION

Here, we outline an example dealing with design of a neurocontroller with NNLM for control of an aircraft in wind shear during take off [3,8]. Comparison of results shows that the dynamic neurocontroller performs well in the presence of wind shear.

We incorporate the same assumptions as Miele and Leitmann, [51,52]:

(1) the rotational inertia of the aircraft and the sensor and actuator dynamics are neglected,
(2) the aircraft mass is constant,
(3) air density is constant,
(4) flight is in the vertical plane,
(5) maximum thrust is used.

We employ equations of motion for the center of mass of the aircraft, in which the kinematic variables are relative to the ground while the dynamic ones are taken relative to a moving but non-rotating frame translating with the wind velocity at the aircraft's center of mass. The kinematic equations are

$$\dot{x} = V\cos(\gamma) + W_x,$$
$$\dot{h} = V\sin(\gamma) + W_h.$$

The dynamical equations are

$$m\dot{V} = T\cos(\alpha+\delta) - D - mg\sin\gamma - m(\dot{W}_x\cos\gamma + \dot{W}_h\sin\gamma), \tag{6.1}$$

$$mV\dot{\gamma} = T\sin(\alpha+\delta) + L - mg\cos\gamma + m(\dot{W}_x\sin\gamma - \dot{W}_h\cos\gamma), \tag{6.2}$$

where $T = T(V)$ is the thrust force, $D = D(h,V,\alpha)$ is the drag, $L = L(h,V,\alpha)$ is the lift, and $W_x = W_x(x,h)$ and $W_h = W_h(x,h)$ are the horizontal and vertical wind shears, respectively. In these equations, $x(t), h(t), V(x), \gamma(t)$ are the state variables and the angle of attack $\alpha(t)$ is the control variable. Other parameters are:

$g \stackrel{\text{def}}{=}$ gravitational force per unit mass, ft sec^{-2};

$m \stackrel{\text{def}}{=}$ aircraft mass, lb $\text{ft}^{-1}\,\text{sec}^2$;

$V \stackrel{\text{def}}{=}$ aircraft speed relative to wind base reference frame, ft sec^{-1};

$\gamma \stackrel{\text{def}}{=}$ relative path inclination, rad;

$\delta \stackrel{\text{def}}{=}$ thrust inclination, rad.

Our goal is to design a control law $\alpha = \alpha_k$ for the discrete-time version of equations (6.1),(6.2) such that the quantity $[h_{k+1} - \hat{h}_r]^2$ is minimized, where h_{k+1} is a value calculated from V_{k+1} and α_{k+1} and $\hat{h}_r$ is a given value for the desired constant rate of climb. The cost function is given by

$$J(k+1) = \left(V_{k+1}\sin\gamma_{k+1} - \hat{h}_r\right)^2.$$

The neural controller is designed so that the weights update at each step to minimize $J(k+1)$.

Wind Shear Model

Much effort has gone into modeling and identifying wind shear, e.g., [53,54]; in this work, we utilize the wind shear model [52] described by the following equations:

$$W_x = -W_{x0}\sin\left(\frac{2\pi t}{T_0}\right),$$

$$W_h = \frac{-W_{h0}\left[1 - \cos\left(2\pi t/T_0\right)\right]}{2},$$

where W_{x0} and W_{h0} are given constants, reflecting the wind shear intensity, and T_0 is the total flight time through the downburst.

Bounded Quantities

In order to account for aircraft capabilities, it is assumed that there is a maximum attainable value of the relative angle of attack α; that is, $\alpha \in [0, \alpha_*]$, where $\alpha_* > 0$. The range of practical values of the relative aircraft speed, V, is also limited, that is,

$$\underline{V} \le V \le \bar{V}.$$

Force Terms

The thrust, drag, and lift force terms can be approximated

$$T = A_0 + A_1 V + A_2 V^2,$$

$$D = \frac{1}{2}C_D\rho S V^2,$$

$$L = \frac{1}{2}C_L\rho S V^2,$$

where $C_D = B_0 + B_1\alpha^2$, $C_l = C_0 + C_1\alpha$. The coefficients A_0, A_1, A_2 depend on the altitude of the runway, the ambient temperature, and the engine power setting. B_0, B_1, C_0, C_1, on the other hand, depend on the flap setting and the undercarriage position.

Controller Design

We employ a neurocontroller with one input layer of four neurons. The input variables to the network are $V, \dot{V}, \gamma, \dot{\gamma}$. The control output is given by $\alpha_k = \phi(w_1(V - V(0)) + w_2\dot{V} + w_3(\gamma - \gamma(0)) + w_4\dot{\gamma})$, where the initial values $V(0)$ and $\gamma(0)$ will be given in the next subsection. The threshold function ϕ is the sigmoidal function $T(x) = A1/(1 + e^{-gx})$, where g is a design gain and A is the saturation limit. In our study $A = \alpha$. The formula for updating the weights is based on a gradient descent algorithm.

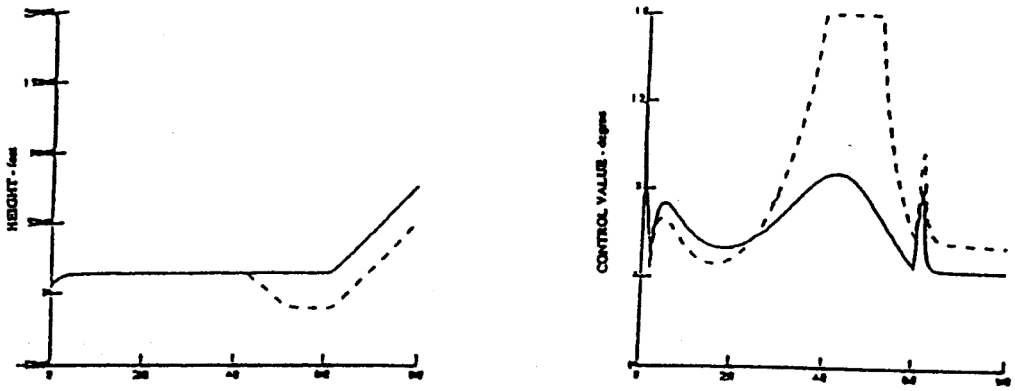
Simulation Results

Numerical simulations were carried out for

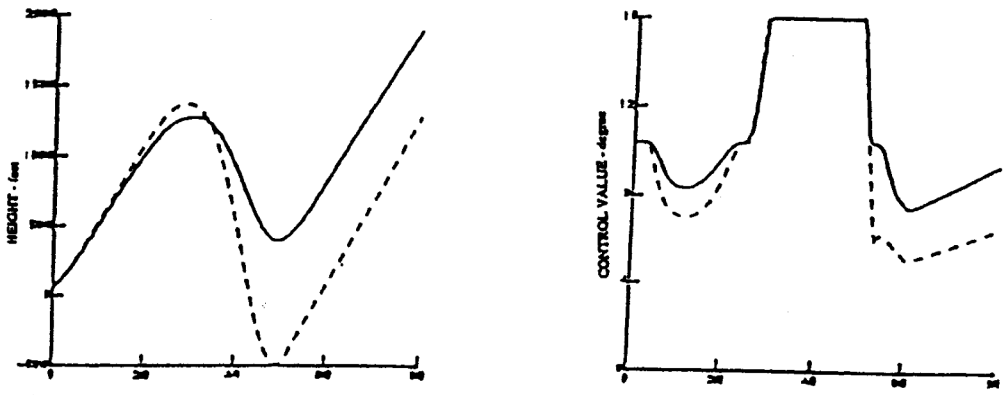
(i) flight in the absence of wind shear using different gains,
(ii) flight with wind shear, again using different gains.

For comparison, we also include the simulation results under the same conditions using the controller proposed by Leitmann, and that of Miele's simplified game guidance (Figures 7a, 7b).

Comparison of the results shows that the neurocontroller performs well in the presence of wind shear. Under the same conditions, the neurocontroller works better than those of Leitmann and Miele. On the other hand, for a more intense wind shear, e.g., $W_{x0}/W_{h0} = 80/48$, we need to adjust the four weights accordingly, since otherwise, the controller will not perform as well.



(a) Miele *et al.*'s controller.



(b) Leitmann's controller.

Figure 7. Comparison of results for control of aircraft in wind shear.

Our attempt in Sections 2–5 of this report has been to mathematically formulate the control systems inside the neural networks. We have represented each linear SISO system in the neural network, by introducing a small feedback loop inside each neuron, rather than a feedback connection. For this paradigm of neural networks, we can directly use the internal states to construct a feedback control law. What is more important is that a network of this type is itself a system, but not an unknown "Black Box". Thus its input-output performance can be studied just as in the case of the classical control system. Based on this observation, many conventional synthesis methods can be directly used to design the system. The stationary property of the system can be preassigned by means of learning, a unique feature that the classical control system does not have.

REFERENCES

1. P.K. De, S.M. Amin and E.Y. Rodin, System identification with dynamic networks, In *Intelligent Engineering Systems through ANN*, Volume 2, (Edited by Dagli, Burke and Shin), pp. 97–102, ASME Press, New York, (1992).
2. Y. Wu, S.M. Amin and E.Y. Rodin, Control and disturbance rejection with a dynamic neurocontroller, In *Intelligent Engineering Systems through ANN*, Volume 2, (Edited by Degali, Burke and Shin), pp. 643–648, ASME Press, New York, (1992).
3. E.Y. Rodin and Y. Wu, On a learning algorithm of feedback control for an aircraft in wind shear: A case study, In *Proceedings of the 31st IEEE Conference on Decision and Control*, Tucson, AZ, (December 1992).
4. E.Y. Rodin and Y. Wu, On neural networks with local memory for control systems, *Appl. Math. Lett.* **4** (5), 97–101, (1991).
5. Y. Wu and E.Y. Rodin, Control systems in neural networks with local memory, *Computers Math. Applic.* **26** (2), 1–24, (1993).
6. Y. Wu, Artificial intelligence methodologies for aerospace and other control systems, Doctoral dissertation, Washington University, St. Louis, MO, (December 1992).
7. S.M. Amin, System identification and disturbance attenuation via dynamic neural networks, Neural Nets for Aero Control Symp., invited paper, NASA Ames Research Center, Moffett Field, CA, (August 1994).
8. S.M. Amin, E.Y. Rodin and Y. Wu, Neurocontrol of an aircraft: Application to windshear, *Mathl. Comput. Modelling* **22** (1), 63–78, (1995).
9. S.M. Amin, E.Y. Rodin, and A.Y. Wu, Application of dynamic neural networks to approximation and control of nonlinear systems, In *Proceedings of the American Control Conferences*, pp. 222–226, Albuquerque, NM, (June 1997).
10. G. Cybenko, Aproximations by superpositions of a sigmoidal function, *Math. Contr., Signals Syst.* **2**, 303–314, (1989).
11. K. Funahashi, On the approximate realization of continuous mappings by neural networks, *IEEE Trans. Neural Networks* **2**, 183–192, (1989).
12. E.B. Kosmatampoulos *et al.*, High-order neural network stuctures for identification of dynamical systems, *IEEE Trans. Neural Networks* **6** (2), 422–431, (1995).
13. H. Khalil, *Nonlinear Systems*, Macmillan, New York, (1992).
14. M. Vidyasagar, *Nonlinear Systems Analysis*, 2nd edition, Prentice Hall, Englewood Cliffs, NJ, (1993).
15. A. Isidori, *Nonlinear Control Systems*, Springer-Verlag, New York, (1989).
16. H. Nijmeijer and A. van der Schaft, *Nonlinear Dynamical Control Systems*, Springer-Verlag, New York, (1991).
17. H. Knobloch, A. Isidori and D. Flockerzi, *Topics in Control Theory*, Birkhäuser Verlag, (1993).
18. W. Miller, R. Sutton and P. Werbos, Editors, *Neural Networks for Control*, MIT Press, Cambridge, MA, (1990).
19. D. White and D. Sofge, Editors, *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches*, Van Nostrand Reinhold, New York, (1992).
20. K. Hunt, D. Sbaraboro, R. Zibkowski and P. Gawthrop, Neural networks for control systems—A survey, *Automatica* **28** (6), 1083–1112, (1992).
21. R. Zbikowski, K. Hunt, A. Dzielinski, R. Murray-Smith and P. Gawthrop, Technical Report ES-PRIT III Project 8039: NACT, Daimler-Benz AG and University of Glasgow, (1994).
22. P. Sastry, G. Santharam and K. Unnikvishnas, Memory neuron networks for identification and control of dynamical systems, *IEEE Trans. Neural Networks* **5** (2), 306–319, (1994).
23. A. Dempo, O. Farotimi and T. Kailath, High-order absolutely stable neural networks, *IEEE Trans. Circuits Syst.* **38** (1), (1991).
24. Y. Kamp and M. Hasler, *Recursive Neural Networks for Associative Memory*, Wiley, New York, (1990).
25. P. Paretto and J.J. Niez, Long term memory storage capacity of multiconnected neural networks, *Biol. Cybern.* **54**, 53–63, (1986).
26. P. Baldi, Neural networks, orientations of the hypercube and algebraic threshold functions, *IEEE Trans. Inform. Theory* **IT-34**, 523–530, (May 1988).

27. O. Farotimi, A Dembo and T. Kailath, Absolute stability and optimal training for dynamic neural networks, In *23rd Annual Asilomar Conference on Circuits, Systems and Computers*, Pacific Grove, CA, Volume 1, pp. 133–137, (Oct. 1989).
28. M. Gori, Y. Benjio and R. Demori, BPS: A learning algorithm for capturing the dynamic nature of speech, IJCNN, Washington, DC, 417–423, (June 1989).
29. M. Gherrity, Learning algorithms for analog, fully recurrent neural networks, IJCNN, Washington, DC, 643–644, (1989).
30. M.J. Willis, C. DiMassimo, G.A. Montague, M.T. Tham and A.J. Morris, Artificial neural networks in process engineering, *IEEE Proceedings, Part D: Control Theory and Applications* **138** (3), 256–266, (May 1991).
31. R. Perfetti, Effect of Feedback in Dynamic Neural Networks, *Electronics Letters* **27** (15), 1367–1369, (July 1991).
32. S.I. Sudharsanan and M.K. Sundareshan, 755–762, *IJCNN*, (June 1990).
33. M. Sato, K. Joe and T. Hirahraa, APOLONN brings us to the real world: Learning nonlinear dynamics and fluctuations in nature, In *Proceedings of IJCNN*, pp. 581–587, San Diego, CA, (June 1990).
34. S. Tan, L. Vandenberghe and J. Vandewalle, Remarks on the stability of asymmetric dynamical neural networks, In *Proceedings of IJCNN*, pp. 461–467, San Diego, CA, (June 1990).
35. T. Mckelvey, Neural networks applied to optimal flight control, In *Selected papers from the IFAC/IFIP/IMACS Symposium, Proceedings of IFAC Symp. on AI in Real-Time control*, (Edited by H.B. Verbruggen and M.G. Rodd), pp. 19–23, Pergamon Press, Oxford, UK, (1992).
36. P.J. Werbos, Neurocontrol and elastic fuzzy logic: Capabilities, concept, and applications, *IEEE Trans. on Industrial Electronics* **40** (2), 170–180, (April 1993).
37. T. Yamaguchi *et al.*, Intelligent control of a flying vehicle using fuzzy associative memory system, In *IEEE Int'l Conference on Fuzzy Systems*, pp. 1139–1149, San Diego, CA, (March 1992).
38. L. Ljung, *System Identification—Theory for the User*, Prentice Hall, Englewood Cliffs, NJ, (1987).
39. P.D. Wasserman, *Advanced Methods in Neural Computing*, Van Nostrand, Reinhold, NY, (1993).
40. S.A. Billings *et al.*, Properties of neural networks with applications to modelling nonlinear dynamical systems, *Int'l. Journal and Control* **55** (1), 193–224, (1992).
41. K.S. Narendra and K. Parthasarathy, Identification and control of dynamical systems using neural networks, *IEEE Transactions on Neural Networks* **1** (2), 4–27, (March 1990).
42. K.J. Aström and Björn Witternmark, *Adaptive Control*, Addison-Wesley, (May 1989).
43. K.S. Narendra and A.M. Annaswamy, Robust adaptive control, In *Adaptive and Learning Systems: Theory and Applications*, (Edited by K.S. Narendra), pp. 3–33, Plenum Press, New York, (1986).
44. R. Hecht-Nielsen, *Neurocomputing*, Addison-Wesley, (1990).
45. V. Kůrková, Kolmogorov's theorem and multilayer neural networks, *Neural Networks* **5** (3), 501–506, (1992).
46. T. Kailath, *Linear System*, Prentice Hall, Englewood Cliffs, NJ, (1980).
47. H.G. Lee, A. Arapostathis and S.I. Marcus, On the linearization of discrete-time systems, *Int. J. Contr.* **45**, 1803–1822, (1987).
48. M. Nikolaou, Y. You and H. Sarimveis, Process modelling with recurrent neural networks, In *Proceedings of the 1st Conference of Artificial Neural Networks in Engineering*, pp. 595–600, St. Louis, MO, (November 1991).
49. T. Troudet, S. Garg, D. Mattern and W. Merrill, Towards practical control design using neural computation, In *Proceedings of the IJCNN*, Volume II, pp. 675–681, (1991).
50. J.W. Grizzle, *Lecture Notes on Control and Information Science*, Volume 83, pp. 273–281, Springer-Verlag, (1986).
51. A. Miele, T. Wang, C.Y. Tzeng and W.W. Melvin, Presented at the *AIAA Guidance, Navigation and Control Conference*, Paper No. AIAA-87-2341, (1987).
52. G. Leitmann and S. Pandey, Presented at the *Conference on Modeling and Control of Uncertain Systems*, (September 1990).
53. W. Frost and D.W. Camp, FAA Report FAA-RD-77, (1977).
54. S. Zhu and B. Etkin, Fluid dynamic model of a downburst, UTLAS Report #271.
55. G.A. Rovithakis *et al.*, Adaptive control of unknown plants using dynamical neural networks, *IEEE Trans. on SMC* **24** (3), 400–412, (March 1994).
56. B. Horne, M. Jamshidi and N. Vadiee, Neural networks in robotics: A survey, *J. of Intelligent and Robotic Systems* (3), 51–66, (1990).
57. S. Lee and G.A. Bekey, *Application of Neural Networks to Robotics Control and Dynamic Systems*, Volume 39, pp. 1–69, Academic Press, New York, (1991).
58. D. Rumelhart and J. McClelland, *Parallel Distributed Processing*, Volume 1: Foundations, MIT Press, Cambridge, MA, (1986).
59. J. Hertz, A. Krogh and R. Palmer, *Introduction to the Theory of Neural Computation*, Addison-Wesley, (1991).
60. E. Tzirkel-Hancock and F. Fallside, Technical report, Cambridge University, (1991).
61. A. Levin and K.S. Narendra, Control of nonlinear dynamical systems using neural networks: Controllability and stabilization, *IEEE Trans. on Neural Networks* **4** (2), 192–206, (March 1993).
62. T. Kohonen, An introduction to neural computing, *Neural Networks* **1** (1), 3–16, (1988).
63. J.J. Hopfield, Neural computations of decisions in optimization problems, *Biological Cybernetics* **52**, 14–152, (1985).

64. J.J. Hopfield and D. Tank, Computing with neural circuits: A model, *Science* **233**, 625–733, (1986).
65. D. Tank and J.J. Hopfield, Simple neural optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit, *IEEE Transaction on Circuits and Systems* **33** (5), 533–541, (1986).
66. M. Nikolaou and V. Hanagaudi "Input-output exact linearization of nonlinear dynamical systems modelled by recurrent neural networks, In *Proceedings of the* 1^{st} *Conference of Artificial Neural Networks in Engineering*, pp. 575–579, St. Louis, MO, (November 1991).
67. J.J. Hopfield, Neural networks and physical systems with emergent collective computational abilities, In *Proceedings Nat. Acad. Sci.*, U.S., Volume 79, pp. 2554–2558, (April, 1982).
68. H.G. Lee and S.I. Marcus, Approximate and local linearizability of nonlinear discrete-time systems, *Int. J. Contr.* **44**, 1103–1124, (1986).
69. K.J. Aström, Theory and applications of adaptive control—A survey, *Automatica* **19**, 471–481, (1983).
70. G.C. Goodwin and K.S. Sin, *Adaptive Filtering Prediction and Control*, Prentice Hall, Englewood Cliffs, NJ, (1984).
71. J. Hopfield, Neurons with graded response have collective computational properties like those of two-state neurons, *Proc. Natl. Acad. Sci.* **81**, 3088–3092, (1987).
72. P. Ioannou *et al.*, Two algorithms for nonlinear system identification using dynamical neural networks, *IEEE ACC* (submitted), (1991).
73. B. Fernandes, Nonlinear dynamic system identification using artificial neural networks: The discrete time case, *IEEE CDC*, (1990).