

A Neural Network Approach to Feedback Linearization

by

Khaled A. S. Al-Sahli

A Thesis Presented to the

FACULTY OF THE COLLEGE OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

ELECTRICAL ENGINEERING

March, 1995

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
313/761-4700 800/521-0600



**A NEURAL NETWORK APPROACH TO FEEDBACK
LINEARIZATION**

BY
KHALED A. S. AL-SAHLI

A Thesis Presented to the
FACULTY OF THE COLLEGE OF GRADUATE STUDIES
KING FAHD UNIVERSITY OF PETROLEUM & MINERALS
DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

ELECTRICAL ENGINEERING

MARCH 1995

UMI Number: 1375310

UMI Microform 1375310
Copyright 1995, by UMI Company. All rights reserved.

**This microform edition is protected against unauthorized
copying under Title 17, United States Code.**

UMI
300 North Zeeb Road
Ann Arbor, MI 48103

**KING FAHD UNIVERSITY OF PETROLEUM AND MINERALS
DHAHRAN 31261, SAUDI ARABIA**

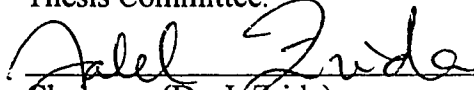
COLLEGE OF GRADUATE STUDIES

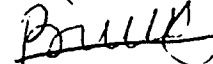
This thesis, written by

KHALED A. S. AL-SAHLI

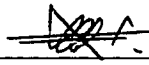
under the direction of his Thesis Advisor and approved by his Thesis Committee, has presented to and accepted by the Dean of the College of graduate studies, in partial fulfillment of the requirements for the degree of MASTER OF SCIENCE. IN ELECTRICAL ENGINEERING.

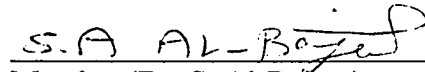
Thesis Committee:

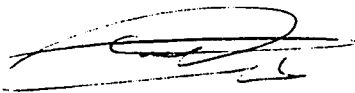

Chairman (Dr. J. Zrida)


Member (Dr. M. Bettayeb)

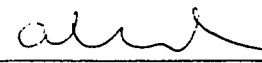

Member (Dr. H. Baher)


Member (Dr. J. Bakhawain)


Member (Dr. S. Al-Baiyat)



Dr. A. Al-shehri
Department Chairman


Dr. Ala H. Rabeh
Dean, College of Graduate Studies

Date: 9.4.95



**A NEURAL NETWORK APPROACH TO FEEDBACK
LINEARIZATION**

K. A. S. AL-SAHLI

ELECTRICAL ENGINEERING

1995

Dedicated to

My Parents, My Wife and My Son Talal

ACKNOWLEDGMENT

Praise be to the Lord of the world, the Almighty for having guided me at every stage of my life.

Words are not adequate to describe my gratitude towards my advisor Dr. J. Zrida. It was a real pleasure working with him. I would also like to place on record my appreciation for the cooperation and guidance extended by my committee members, Dr. M. Bettayeb, Dr. H. Baher, Dr. J. Bakhashwain, And Dr. S. Al-Baiyat.

I also acknowledge the generous help and support for this research given by KFUPM. My thanks also to the chairman, faculty and my friend Mr. M. Obaid.

Contents

Acknowledgment	i
List of Figures	v
List of Tables	viii
Nomenclature	ix
Abstract (English)	1
Abstract (Arabic)	1
1 INTRODUCTION	2
1.1 Motivation.....	2
1.2 Organization of the work	4
2 LITERATURE SURVEY	5
3 NEURAL NETWORK	11
3.1 Artificial neuron	12
3.2 Multilayer Neural Network	14

3.3 Backpropagation Training Algorithm	17
3.4 Training the Network	24
4 FEEDBACK LINEARIZATION	26
4.1 Mathematical Tools	27
4.2 Input to state feedback linearization	30
4.3 State Transformation	35
4.3.1 Method 1.....	35
4.3.2 Method 2.....	37
4.3.3 Method 3.....	37
4.3.4 Method 4.....	39
4.4 Limitations of Feedback linearization	40
5 FEEDBACK LINEARIZATION USING NEURAL NETWORK	42
5.1 Motivation.....	42
5.2 Basic Concepts And Definitions.....	44
5.3 Problem Statement.....	45
5.4 Stabilization Problem.....	46
5.5 Feedback Linearization Using Neural Networks.....	49
5.5.1 Scheme 1.....	49
5.5.2 Scheme2.....	54
6 SIMULATION RESULTS	58
5.1 Antenna Arm Control System	59
5.2 Inverted pendulum Stabilization Problem	72
5.3 Van Der Pol Equation	88

7 CONTRIBUTIONS, CONCLUSIONS AND RECOMMENDATIONS

6.1 Contributions	99
6.2 Conclusions	100
6.3 Recommendations	100
 BIBLIOGRAPHY	 102
 Vita	 107

List of Figures

Fig		page
2.1	Architecture for feedback linearization	8
2.2	Architecture for estimation of the plant	9
3.1	Structure of Artificial Neuron	13
3.2	Structure of Multilayer Neural Network with one hidden layer	15
3.3	Structure of Multilayer Neural Network with 2 hidden layers	15
3.4	Structure of Single Neuron	16
3.5	Architecture for approximation of nonlinear function	17
3.6	A network example to derive BEP	18
3.7	Structure of BP through a plant	21
3.8	An alternative solution using NN to learn the inverse dynamics	23
3.9	An alternative solution using indirect adaptive control method	24
5.1	Block Diagram for scheme 1	50
5.2	Block diagram of feedback linearization controller using NN	54
6.1	Antenna Arm Control System	59
6.2	NN controller response and the perfect cancellation controller	65
6.3a	The first linear and linearized state with the initial condition: $x_1 = -45^\circ$, and $x_2 = 0$	66
6.3b	The error between the first linear and linearized states	67
6.3c	The second linear and linearized state with the initial condition: $x_1 = -45^\circ$, and $x_2 = 0$	68
6.3d	The error between the second linear and linearized states	69
6.3e	The linearizing input calculated by feedback linearization method and the linearizing input generated by the proposed NN control	70

6.3f	The error between the linearizing input given by Feedback linearization method and the linearizing input produced by NN	71
6.4	Inverted pendulum controlled by a DC motor	73
6.5	Model of an armature-controlled DC motor	73
6.6	Response of uncontrolled inverted pedulum	75
6.7a	The first linear and linearized state with initial condition: $x_1 = 45^\circ$, and $x_2 = x_3 = 0$	80
6.7b	The error between the first linear and linearized state	81
6.7c	The second linear and linearized state with initial condition: $x_1 = 45^\circ$, and $x_2 = x_3 = 0$	82
6.7d	The error between the second linear and linearized states	83
6.7e	The third linear and linearized state with initial condition: $x_1 = 45^\circ$, and $x_2 = x_3 = 0$	84
6.7f	The error between the third linear and linearized states	85
6.7g	he linearizing input calculated by feedback linearization method and the linearizing input generated by the proposed NN control	86
6.7h	The error between the linearizing input given by Feedback linearization method and the linearizing input produced by NN	87
6.8	The limit cycle of the system	89
6.9a	Stabilization of Van Der Pol at zero position with the proposed NN control with initial condition: $x_1 = 1$, and $x_2 = 0$.The first states of the linear and linearized system	93
6.9b	The error between the first linear and linearized state	94
6.9c	The second linear and linearized state with initial condition: $x_1 = 1$, and $x_2 = 0$	95
6.9d	The error between the second linear and linearized state	96

6.9e	The linearizing input calculated by feedback linearization method and the linearizing input generated by the proposed NN control	97
6.9f	The error between the linearizing input given by Feedback linearization method and the linearizing input produced by NN	98

List of Tables

Table	page
1 NN configuration used for the control of Antenna arm control system	63
2 NN configuration used to control an inverted pendulum system	79
3 NN configuration used to control Van Der Pol nonlinear system	91

Nomenclature

NN	neural network
LRM	linear reference model
x	actual nonlinear state
x_m	desired state
$y_{a,p}$	actual or plant output
r	reference command
u	input signal
e	output error
δ	backpropagated error
η	learning rate
$H(\cdot)$	activation function
ε	cost function to be minimized
w_{ij}^s	weight connecting the i th neuron at layer s to the previous layer's neuron j
∇	the gradient
$T(\cdot)$	nonlinear transformation
$\alpha(\cdot)$	nonlinear feedback transformation
$\beta(\cdot)$	linear transformation in the input
$f(\cdot), g(\cdot)$	differentiable nonlinear functions
TDL	delay

خلاصة الرسالة

اسم الطالب: خالد عطا آلّة سليم السهلى

عنوان الرسالة: استخدام الشبكات العصبية الصناعية في قضية تحويل الانظمة الغير خطية الى انظمة خطية بطريقة التغذية المرتدة.

التخصص: الهندسة الكهربائيه

تاريخ التخرج: شوال ١٤١٥هـ.

ظهرت اخيرا طرق عديدة لتصميم و تحليل التحكم فى الانظمة الغير خطية غير ان كل طريقة يفضل استخدامها لصنف معين من هذه الانظمة. وتعتبر طريقة التحكم بتحويل الانظمة الغير خطية الى انظمة خطية باستخدام التغذية المرتدة واحدة من افضل الطرق الموجوده الا ان لهذه الطريقة عيوب كثيره ولاستخدم الا للانظمة القابلة لتحويلها خطيا. ولقد ظهر فى السنوات الاخيرة اهتمام ملحوظ بالشبكات العصبية الصناعيه, وتعد انظمة التحكم ذات التوجيه الذاتى واحده من تطبيقاتها. فى هذه الدراسه تم تطوير بسيط للطريقه المستخدمه فى معالجة الاعصاب الصناعيه التى تربط الخلايا العصبية الصناعيه ببعضها والتى تسمى طريقة التغذية المرتدة للخطأ. كما تم فى هذه الدراسه استخدام الشبكة العصبية الصناعيه ذات المستويات العديده للتحكم بالانظمة الغير خطية تحكم مباشر و ذاتي التوجيه لتلافي عيوب الطريقة المذكورة وتم استخدام المحاكاة لدراستها.

درجة الماجستير فى العلوم

جامعة الملك فهد للبترول والمعادن

الظهران-المملكه العربيه السعوديه

شوال ١٤١٥هـ

Abstract

Name: Khaled A. S. Al-sahli
Title: A Neural Network Approach to Feedback Linearization
Major Field: Electrical Engineering
Date of Degree: 1995

Several methods are available for the design and analysis of nonlinear control systems. However, each method is best applicable to a special class of nonlinear systems. The Feedback linearization scheme is a powerful method, because it is based on exact cancellation of nonlinearities. This method is, however, only applicable to systems which are feedback linearizable. It also has a number of limitations. In recent years, there has been a considerable interest in the field of artificial neural networks and its applications in many areas. One of the application areas is in adaptive control systems. In this thesis, the conventional backpropagation algorithm is modified in such a way that nonlinear systems can be controlled directly. In this work, a multilayer neural network is used to perform input-state feedback linearization thus alleviating its drawbacks for a large class of nonlinear systems, adaptively and in an on-line manner. A simulation study supports this claim.

Master of Science Degree
King Fahd University of Petroleum and Minerals
Dhahran, Saudi Arabia
1995

CHAPTER 1

INTRODUCTION

1.1 Motivation

In this chapter a brief look at the motivation behind the research in the field of nonlinear systems analysis and design is taken. The motivation of using neural networks for feedback linearization is also established. Finally, the organization of the thesis is presented.

Previously, the computational difficulty associated with nonlinear control analysis and design has limited the application of nonlinear control methods. This is, perhaps due to the complex behavior of nonlinear systems which can display many, often undesirable, phenomena, such as limit cycle, finite escape time, and chaos phenomena. Besides, nonlinear systems are described by nonlinear differential equations which cannot be solved analytically. Also, powerful mathematical tools like Laplace and Fourier transform do not apply to nonlinear systems.

As a result, there are no systematic tools for analyzing and designing nonlinear control systems. In the past, feedback control systems were restricted to linear systems or

systems that can be linearized by conventional linear methods, such as the traditional linearization process which expands the nonlinear differential state equations into a Taylor series about an operating point. Although, linear control methods perform well over a small range of operation, they are impractical to physical systems which are nonlinear to some degree.

Recently, as the advent of powerful microprocessors improved our computational ability, many researchers and designers renewed their interest in the development and application of nonlinear control methodologies. Nowadays, tools for analyzing and designing nonlinear control systems are available, such as the describing function approach, the phase plane method and Lyapunov's methods, and adaptive control. Each one of these approaches is, however, only applicable to a particular class of nonlinear systems.

In spite of these methods that have been proposed, our understanding of the behavior of nonlinear systems is still far from complete, and the linearization technique is still of great interest. Unfortunately, the available methods for linearization of nonlinear systems such as approximate linearization, and pseudolinearization are based on approximations of the dynamics.

Recently, the feedback linearization method has been proposed and successfully used to control some practical problems in many areas, that include aerospace engineering, power systems, robotics, and chemical processes. This linearization technique has the unique feature that it is achieved by exact cancellation of the nonlinear terms through an algebraic state transformation and feedback, rather than by approximations. The linearization of nonlinear systems using feedback linearization method does not, however, guarantee robustness. Consequently, if there are errors or uncertainty in the model, the cancellation does no longer take place. In addition, this

technique performs linearization in an off-line manner, so it requires a prior knowledge of the nonlinear system model.

In recent years, advances in the area of artificial neural networks (ANN) have provided new insights into the control of systems with complex, unknown, and nonlinear dynamics. The idea of using neural networks to perform feedback linearization is not new and many authors reported works along these lines [2,12,13,14]. Their approaches, however, are restricted to simply approximating the nonlinear terms in the dynamics by neural networks. In this thesis, neural networks are used, for the first time as feedback linearization controllers. The motivation behind using neural networks in this approach is to perform feedback linearization by generating the linearizing input required for the linearization adaptively and in an on-line manner for a large class of nonlinear systems, not necessarily in the controllability canonical form. In essence, it is used to remove some of the feedback linearization method drawbacks.

1.2 Organization of the Work

Chapter 2 deals with the literature review of feedback linearization. Chapter 3 provides a brief review of neural networks and their structure and how they are trained. The basis of feedback linearization is briefly discussed in chapter 4. Chapter 5 includes the development of feedback linearization using neural network. The simulation results are provided in Chapter 6. Chapter 7 includes conclusion as well as suggestion for future work.

CHAPTER 2

LITERATURE SURVEY

A survey of the literature on feedback linearization is provided in this chapter. Also, we look at the recent trends in the utilization of neural networks for the purpose of nonlinear control.

Feedback linearization was implicitly or explicitly discussed in several papers dealing with the study of noninteracting control of nonlinear systems [37,41].

In [1], Krener solved the problem in which nonlinear systems can be transformed into linear systems, using only a change of variables. Brockett [43], gave sufficient conditions for a real nonlinear system with an equilibrium point at the origin to be locally equivalent to a linear system in integrator form, using change of coordinates and feedback. Starting with the work of Brockett, several authors studied the problem of when a differential equation relating the input to the state can be linearized via state feedback and coordinate transformation. The problem was completely solved by Jackubczyk and Respondek in [4]. Also, independently, it was solved by Hunt, Su, and Meyer in [35], where necessary and sufficient conditions for the existence of a local transformation which carries a nonlinear system to a linear system in the controllability

canonical form are given. The transformation is based on both feedback and change of coordinates and it is more general than those found in [43]. A global transformation in the whole state space procedure is developed in [27].

Nowadays, there are different transformation methods available for nonlinear systems [39]. These transformation schemes are basically used to transform nonlinear systems into the controllability canonical form. Recently, a transformation into generalized controller canonical forms for nonlinear dynamics is proposed in [30]. Under such transformation, all nonlinear dynamics can be exactly linearized via dynamic feedback.

The theory of feedback linearization is now well developed and understood [49]. Also, Slotine [19] has given a complete discussion of this theory and its drawbacks for single input and single output (SISO) nonlinear systems, and for multi input-multi output (MIMO) nonlinear systems. Conditions under which nonlinear systems can be linearized have also been discussed.

Feedback linearization has been successfully used in a number of practical applications. These include: robot manipulators [44], aerospace systems [45], chemical processes [46], power systems [38], and DC motors [17,33].

An excellent survey of feedback linearization and other linearization methods for nonlinear systems, and also the future areas for research have been given in [11].

The utilization of neural networks in the field of control systems is expanding rapidly. There is a significant research effort in this direction. This can be inferred from the growth of papers, journals, conferences, and conference sessions devoted to the topic of Artificial Neural Networks (ANN), and their application just in the control systems area.

The basic ideas and techniques of Artificial Neural Networks are well explained in [3,5,9,16,36].

In the literature on neural network architectures for control, a large number of control structures have been proposed and used with the backpropagation algorithm as its learning rule. In [3,10,36], training algorithms for dynamics plants with applications to controlling different nonlinear systems are described. In [10], three learning architectures are proposed for training the neural controller to provide the desired inputs to the plant so that a desired response is obtained. Also, a modified backpropagation algorithm based on propagation of the output error through the plant is introduced. This proposed method avoids the training stage, and the network learns continuously and is therefore adaptive. However, the evaluation of the output error requires a prior knowledge of the plant. Psaltis et al. [10], consider that the plant can be thought of as unmodifiable additional layer of the neural controller.

Recently, Narrendra et al. [24] have shown by simulations that a neural network can be used effectively for the identification and control of nonlinear dynamic systems depending on several assumptions regarding the input-output behavior of the plant (e.g. complete controllability and observability are assumed). The results are based on the universal approximation properties of multilayer neural networks provided in [21].

In [42], a modified backpropagation algorithm through the plant is proposed. Yet, the modification is based on emulating the unknown plant. In [31], a neural controller based on backpropagation algorithm is proposed. Although, the weights are adapted directly in terms of the output of the plant, the partial derivatives which are unknown are approximated by their sign.

The idea of applying neural network to control nonlinear feedback linearizable systems appeared in [12,13,14,15], however, the proposed approaches are based only on using the neural network to approximate the nonlinear terms.

An excellent survey of the importance of neural networks from a control systems perspective and also the future areas for research have been given in[22]. The main focus is on the promise of Artificial Neural Networks in the realm of modeling, identification, and control of nonlinear systems.

In [2], Levin and Narrendra suggested a method of using neural network for feedback linearization. Three different multilayer neural networks with different learning algorithms are used to approximate the nonlinear functions, the state transformations, and the linearizing input. In this approach, the plant's dynamics are approximated using an identification method which uses a neural network with a static backpropagation algorithm. After that, two distinct neural networks are trained simultaneously to approximate the transformation and the nonlinear feedback controller using a static and a dynamic backpropagation algorithm, respectively. Also, the weights are adjusted over k -steps. As given in [2], the overall block diagram for feedback linearization using neural networks is illustrated in fig 2.1.

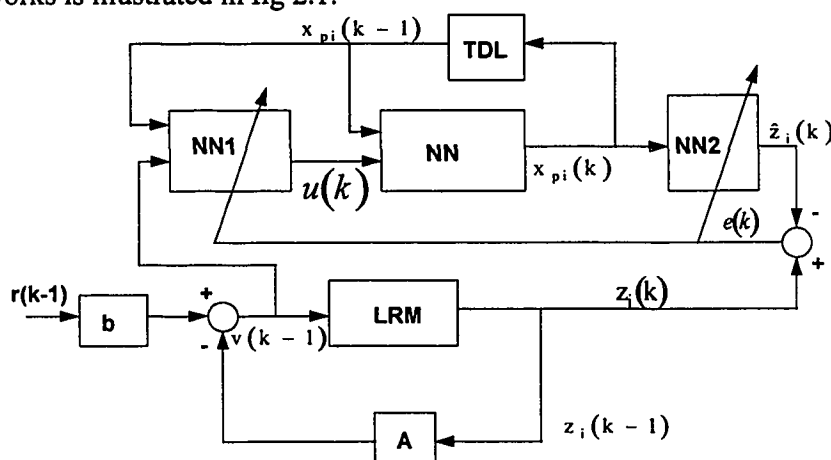


Figure 2.1 Architecture for feedback linearization

An essential first step in this method is to emulate the nonlinear plant, whether its dynamics are known or unknown. The architecture for emulating the plant is shown in fig. 2.2.

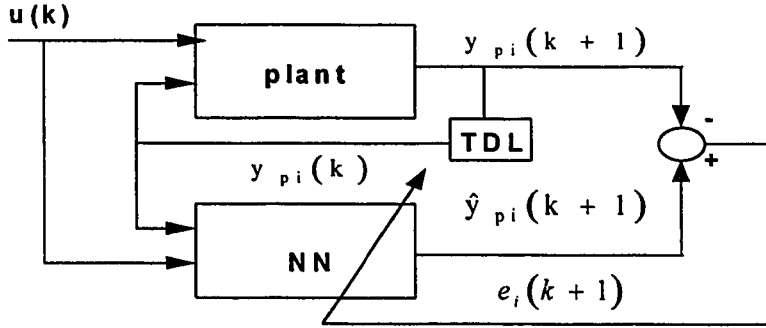


Figure 2.2 Architecture for estimation of the plant

After the emulation step, NN1 and NN2 in fig. 2.1 are trained simultaneously. The weights of NN2 are adjusted using a static backpropagation algorithm, while the weights of NN1 are adjusted by a dynamic backpropagation scheme. The dynamic backpropagation algorithm used to update the weights of NN1 in fig. 2.1 is derived as follows:

If the instantaneous error $e(k)$ in fig. 2.1 is given by:

$$e(k) = z(k) - \hat{z}(k)$$

where z and \hat{z} are the desired output and the neural network output respectively, also y and \hat{y} in fig. 2.2 represent the desired output and the approximated one respectively.

The performance criterion over an interval can be characterized by

$$E = \sum_k \|z(k) - \hat{z}(k)\|^2 \equiv \sum_k \|e(k)\|^2$$

Calculating the gradients of the performance criterion with respect to the weights of NN2 is derived as follows:

$$\frac{dE}{dw} = -2 \sum_k [z(k) - \hat{z}(k)]^T \frac{d\hat{z}(k)}{dw}$$

and

$$\frac{d\hat{z}(k)}{dw} = \sum_j \frac{\partial \hat{z}(k)}{\partial x_j(k)} \frac{dx_j(k)}{dw}$$

where j represents the overall neurons in the layer. The last derivative $\frac{dx_j(k)}{dw}$ is given

by:

$$\frac{dx_j(k)}{dw} = \sum_l \frac{\partial x_j(k)}{\partial x_l(k-1)} \frac{dx_l(k-1)}{dw} + \frac{\partial x_j(k)}{dw}$$

where $w \in W(\text{NN2})$, and l is the overall neurons in the previous layer.

However, the above dynamic backpropagation is computationally not efficient. In this thesis, a simpler computationally efficient method is proposed. In this work, we only use one neural network which serves as a direct adaptive feedback linearization controller, and the weights are adjusted at each instant of time rather than over k -steps using a static backpropagation algorithm. This proposed scheme is performed on-line and with no training stage. Although, the backpropagation through the plant algorithm is used, the unknown partial derivative is approximated by its real value instead of approximating the unknown partial derivative by its sign. The idea of using the neural network as a direct adaptive feedback linearization controller is not found in published papers.

The issue of utilizing neural network in control systems continues to be a hot topic of research these days as witnessed in part by recent special issues on the subject IEEE control system magazine [9].

CHAPTER 3

NEURAL NETWORKS

In this chapter the fundamentals of neural networks are described. The architecture of the multilayered neural networks is also included. A modified backpropagation algorithm is discussed at the end of this chapter.

Neural networks are computational systems, either hardware or software, which mimic the computational abilities of biological systems by using a large number of simple interconnected artificial neurons. Artificial neurons are simple emulation of biological neurons; they take in information from sensors or other artificial neurons, perform simple operations on this data and pass the results on to other neurons. Neural networks can be described as an attempt to achieve human-like performance. It would be more suitable to describe it as an attempt to mimic the way the brain does things. In this respect, the neural network structure is based on our current understanding of the biological nervous system.

3.1 Artificial Neuron

The fundamental cell of an artificial neural network is the artificial neuron. The structure of single neuron is shown in figure 3.1. A set of inputs are applied to the neuron over weighted links. The neuron performs a simple operation on the received data. First, it sums up all the weighted inputs and then passes the results through a nonlinear function H . The output of the i th neuron can be expressed as

$$y_i = H\left(\sum_{j=0}^n w_{ij}x_j\right) \quad 3.1$$

Where n is the total number of inputs.

The function H is called the neuron activation function, which could have different forms such as the hard limiter, the threshold logic function, and the nonlinear sigmoid function. The most often used activation function is the nonlinear sigmoid function usually taken to be:

$$H(x) = \frac{1}{1 + e^{-x}} \quad 3.2$$

but, we will use the tangential sigmoid function in this thesis, usually taken as:

$$H(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \quad 3.3$$

One of the advantages of the tangential sigmoid function is that it has a simple derivative:

$$H'(x) = (1 - H^2(x)) \quad 3.4$$

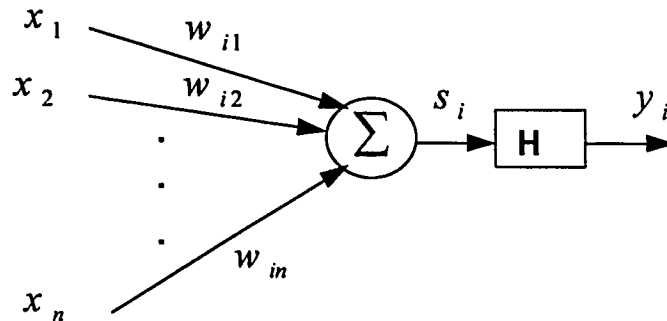


Figure 3.1 Structure of artificial neuron

Regardless the variety of neural network models, all are based on the artificial neuron configuration. There are six basic typologies of neural networks. These are :

1. Multilayered feedforward networks.
2. Single layer, laterally connected.
3. Single layer, topologically ordered
4. Bilayer, feedforward/feedback networks.
5. Multilayer cooperative network
6. Hybrid network

In this thesis, we will adopt the multilayered feedforward neural network which will be introduced in the following section.

3.2 Multilayer Neural Network (MNN)

The multilayer neural networks (mnn) are feedforward networks with one or more layers between the input and output layers. Neurons in each layer are fully connected to

only the neurons in the next layer. Because of the great capabilities offered by the multilayer neural network, they overcome many of the single layer limitations.

The capabilities of multilayer neural network are enhanced by the nonlinearities used. Nonlinearities must be present, because without it the multilayer would implement nothing more a linear transformation in which case the multilayer neural network could be reduced to an equivalent single layer network.

In practice, the multilayer neural networks have been successfully used in many different areas. Multilayer networks have good potential for control applications, because they have the ability to learn and can approximate a wide range of nonlinear functions to any desired degree of accuracy.

A neural network consists of a set of interconnected neurons. In multilayer feedforward neural network, neurons are organized in layers $s=0,1\dots S$ and a neuron in layer s receives its inputs only from neurons in the $s-1$ layer. A typical multilayer neural network with an input layer, an output layer and one hidden layer is shown in figure 3.2, and with two hidden layers in fig. 3.3. In common practice, the input layer is referred to as the zeroth input layer, i.e. $s=0$ is the input layer, and to the $s=S$ layer as the output layer and all others as hidden layers.

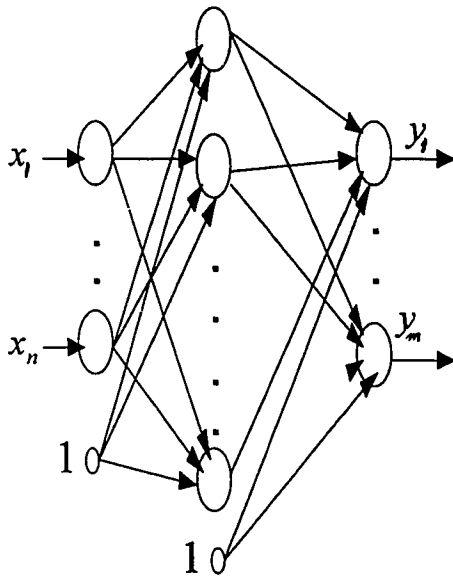


Figure 3.2 Structure of Multilayer
with one hidden layer

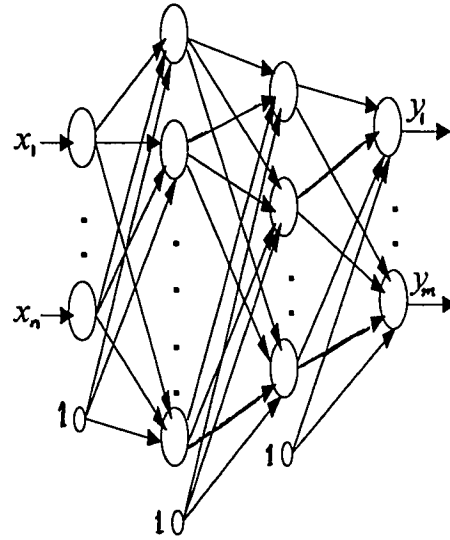


Figure 3.3 Structure of Multilayer
with two hidden layers

The structure of single neuron in both the hidden and the output layers is illustrated in fig. 3.4. The output of the i -th neuron in layer s is given by [9]:

$$y_i^s = H\left(\sum_{j=0}^n w_{ij}^s x_j^{s-1}\right) \quad 3.5$$

where w_{ij}^s is the weight connecting the i -th neuron at the s -th layer to the previous layer's neuron j . The biases w_{i0}^s are treated as additional weight from a neuron with output always at one. In this work, the output of the neuron at the output layer is chosen to be linear, that is:

$$y_i^s = \sum_{j=0}^n w_{ij}^s x_j^{s-1} \quad 3.6$$

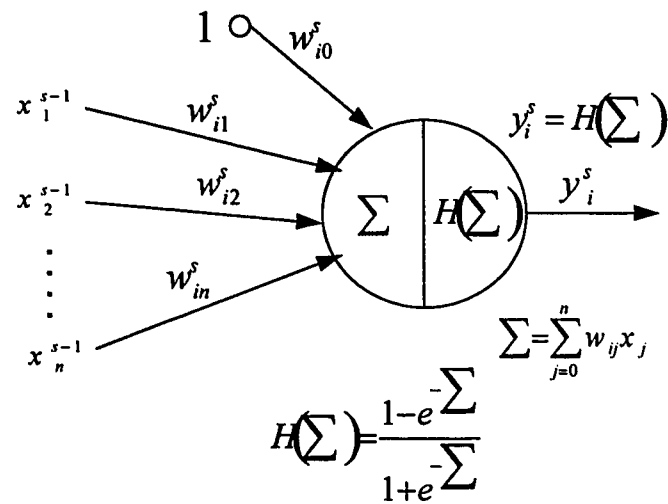


Figure 3.4 Structure of single neuron

It was proved in [21] that the multilayer neural network with one hidden layer contains large number of neurons can approximate any continuous nonlinear function to the desired degree of accuracy. In control system, a neural network is used to approximate a continuous nonlinear function as shown in fig. 3.5. Here, the weights of the neural network will be adjusted such that the output of NN matches the desired output. If the nonlinear function is known, then a neural network NN can be trained off-line. When the function to be approximated is unknown, a neural network NN has to be trained on-line using the input-output pairs.

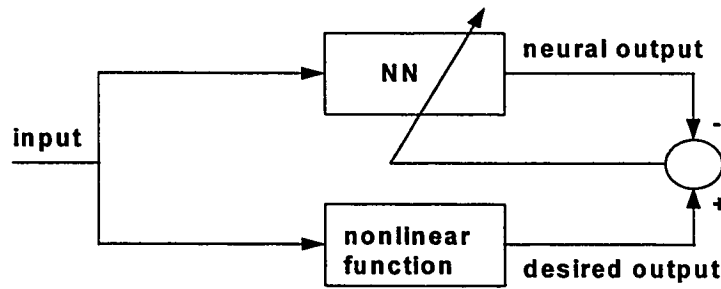


Figure 3.5 Architecture for approximation of nonlinear function

In the following section, we will explain how to adjust the weights of the neural network, so the backpropagation training algorithm will be introduced.

3.3 Backpropagation Training Algorithm (BEP)

Much of the present interest in neural networks has to do with their ability to learn. The most commonly used learning algorithm in neural networks is based on error backpropagation used for adjusting the interconnection weights between layers. The multilayer neural network is trained to map the input sets into points in the range corresponding to output classes. The backpropagation algorithm is a gradient descent technique. It is used mainly to find the network weights that minimize a criterion function. The criterion function to be minimized is the sum of the squared error between the desired output and the network actual output. That is to say, the backpropagation scheme adjusts the weights of the network in the direction opposite to the instantaneous error gradient defined as [3]:

$$\nabla = \frac{\partial E}{\partial w} \quad 3.7$$

The sum squared error E , is the criterion function to be minimized. It is defined as the sum of the square of the errors at each neuron at the output layer, where the output

error is the difference between the desired and the actual responses. The output error can be expressed in general as:

$$e_i = y_{mi} - y_i \quad 3.8$$

Where y_{mi} stands for the desired response of the neuron i , and y_i is the actual output of the same element. Also, the instantaneous sum squared error for the s -th layer is given by:

$$E = \frac{1}{2} \sum_{i=1}^n e_i^2 \quad 3.9$$

Where n is the total number of neurons in the s -th layer.

Implementation of the backpropagation algorithm starts by presenting an input pattern vector X to the input layer of the network with the initial weight values set to small random numbers. the input pattern is propagated through the network to generate an output vector Y . See fig. 3.6 which zooms on the i -th neuron of a multilayer neural network.

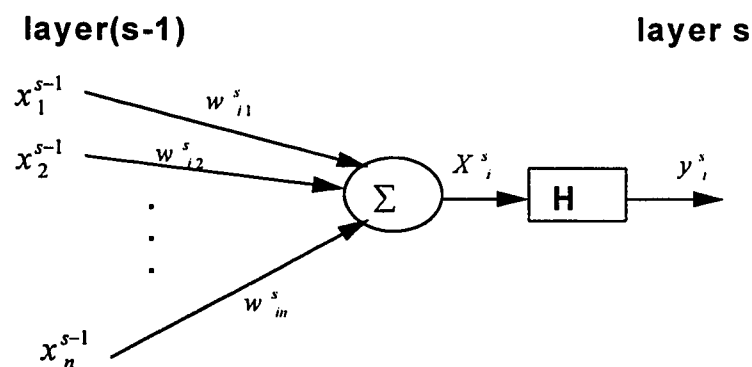


Figure 3.6. A network example to derive BEP

we have,

$$y_i^s = H(X_i^s) = H\left(\sum_{j=1}^n w_{ij}^s x_j^{s-1}\right) \quad 3.10$$

where the w_{ij}^s are the weights connecting neuron i to the previous layer's neuron j , and x_j^{s-1} are the output of the previous layer.

If layer s is the output layer, the criterion function to be minimized in this example is the global error given by the following relation:

$$E = \frac{1}{2} \sum_i (y_{mi} - y_i^s)^2 \quad 3.11$$

where y_{mi} is the desired output of neuron i and y_i^s is the actual output of the same element at layer s . This error will be minimized by the steepest descent scheme in order to adjust the weights by the following relation;

$$w_{ij}^s(k+1) = w_{ij}^s(k) - \eta \frac{\partial E}{\partial w_{ij}^s} \quad 3.12$$

where η is the convergence rate. The partial derivative in the above equation is given by:

$$\frac{\partial E}{\partial w_{ij}^s} = \frac{\partial E}{\partial X_i^s} \cdot \frac{\partial X_i^s}{\partial w_{ij}^s} \quad 3.13$$

then the local error for an output node is given by;

$$e_i^s = - \frac{\partial E}{\partial X_i^s} = - \frac{\partial E}{\partial y_i^s} \cdot \frac{\partial y_i^s}{\partial X_i^s} = (y_{mi} - y_i^s) H'(X_i^s) \quad 3.14$$

and,

$$\frac{\partial X_i^s}{\partial w_{ij}^s} = x_j^{s-1} \quad 3.15$$

Therefore, the weights are adapted by the following relation:

$$w_{ij}^s(k+1) = w_{ij}^s(k) + \eta e_i^s x_j^{s-1} \quad 3.16$$

However, if Layer s is a hidden layer, then the local error is determined entirely by the local error at the next layer and is given by the following relation;

$$e_i^s = -\frac{\partial E}{\partial X_i^s} = -\frac{\partial E}{\partial y_i^s} \cdot \frac{\partial y_i^s}{\partial X_i^s} \quad 3.17$$

In this case,

$$-\frac{\partial E}{\partial y_i^s} = -\left(\frac{\partial E}{\partial X_i^{s+1}} \cdot \frac{\partial X_i^{s+1}}{\partial y_i^s} + \dots + \frac{\partial E}{\partial X_k^{s+1}} \cdot \frac{\partial X_k^{s+1}}{\partial y_i^s} \right) \quad 3.18$$

$$-\frac{\partial E}{\partial y_i^s} = \sum_k e_k^{s+1} w_{ki}^{s+1} \quad 3.19$$

where k is the overall nodes in the above layer.

Therefore, the local error at the hidden layer is given by:

$$e_i^s = H'(X_i^s) \sum_k e_k^{s+1} w_{ki}^{s+1} \quad 3.20$$

Finally, the weights are adapted by using the law given by equation 3.16

$$w_{ij}^s(k+1) = w_{ij}^s(k) + \eta e_i^s x_j^{s-1}$$

with the local error given by equations 3.14 and 3.20 for an output node and hidden node respectively [3,36].

The conventional backpropagation algorithm mentioned above cannot be used in the case when the plant lies between the neural network and its error signal as shown in figure 3.7.

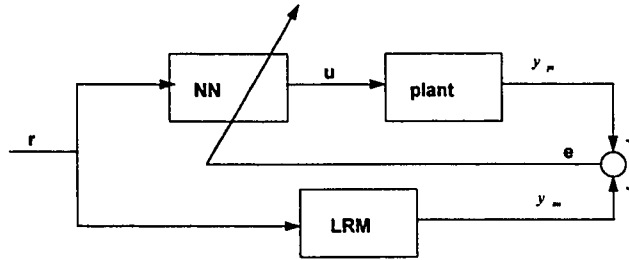


Figure 3.7 Structure of BP through a plant

In this case the backpropagation algorithm has to be modified to adjust the weights by back propagating the output error through the plant to the network. The connecting weights of the network are adjusted by:

$$w_{ij}(k+1) = w_{ij}(k) - \eta \frac{\partial E}{\partial w_{ij}} \quad 3.21$$

Using the chain rule:

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial u} \cdot \frac{\partial u}{\partial w_{ij}} \quad 3.22$$

Since the output layer is chosen to be linear with single neuron in this work, then we have

$$u = \sum_i w_{li} y_{pi} \quad 3.23$$

therefore,

$$\frac{\partial E}{\partial w_{li}} = y_{pi} \frac{\partial E}{\partial u} \quad 3.24$$

Now the term $\frac{\partial E}{\partial u}$ is to be calculated

$$\frac{\partial E}{\partial u} = \frac{\partial E}{\partial y_{pi}} \cdot \frac{\partial y_{pi}}{\partial u} = (y_{mi} - y_{pi}) \frac{\partial y_{pi}}{\partial u} \quad 3.25$$

Where y_{mi} stands for the i -th desired response and y_{pi} is the i -th plant output as shown in fig. 3.7, and $i=1,2,..n$, where n is the number of the states.

The partial derivative $\frac{\partial y_{pi}}{\partial u}$ is not known, in [10], Psaltis et al. suggested that this approximate derivative can be determined either by changing each input to the plant slightly at the operating point and measuring the change at the output, or by comparing changes with previous iterations. In our approach an approximation for the partial derivative will be used to implement the backpropagation algorithm. Our approximation for the unknown partial derivative can be expressed as:

$$\frac{\partial y_{pi}}{\partial u} = \frac{y_{pi}(k+1) - y_{pi}(k)}{u(k+1) - u(k)} \quad 3.26$$

This approximation will be determined at each instant of time to calculate the output error which will be propagated backward to the network so that the weights of the neural network can be adjusted, rather than comparing changes with previous iteration or changing the input slightly at the operating point and measuring the change at the output.

Finally, using the partial derivative approximation, we can adjust the weights using the relation given by equation 3.16:

$$w_{ij}^s(k+1) = w_{ij}^s(k) + \eta e_i x_j^{s-1}$$

For an output node the error e_i for this type of architecture is given by:

$$e_i = f'(y_o) (y_{mi} - y_{pi}) \frac{\partial y_{pi}}{\partial u} \quad 3.27$$

and for a hidden node it is given as:

$$e_i = f'(y_h) \sum_k e_k w_{jk} \quad 3.28$$

The method mentioned above has some advantages: capability to handle system uncertainties, simple architecture and learning ability. However, the main problem of this method is that the neural network control system usually have more than one equilibrium point. This means that system is easy to get stuck at local minima. Besides, the neural network will stop learning when the approximate partial derivative is equal to zero.

In the rest of this section we would like to place down some alternative solution to the problem mentioned above. Since the error signal at the output of the neural network is not available, then using the adaptive control with indirect learning scheme might solve the problem. In this method, the desired output of the LRM are fed to the inputs of the neural network NN as shown in fig. 3.8, then the network will produce an input signal to drive the plant. After that, feeding the actual output of the plant measured during the control process, to the neural network NN1, the network learns the inverse dynamics of the unknown plant and generates an estimation for the required input in order to achieve the actual plant output. The weights of the network are adjusted by propagating the difference between the actual and the estimated input signals using the conventional backpropagation algorithm. Therefore, the problem of approximating the output error to be propagated through the plant will be removed in such method.

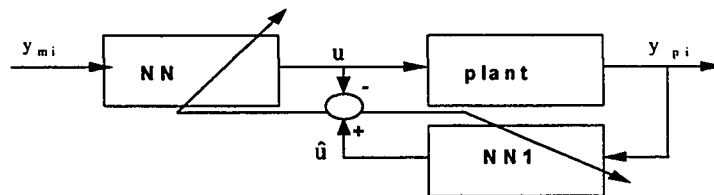


Figure 3.8 An alternative solution using NN to learn the inverse dynamic

As an alternative solution, also we suggest using the indirect adaptive control method which is based on the identification of the plant as shown in fig. 3.9 [24]. Instead of

backpropagating the error through the plant, here the error will be propagated backward through the neural network used to identify the system. The error between the estimated output and the actual ones is used to adjust the weights of the neural network NN1, while the error e_c , is used to adjust the weights of the neural network NN by propagating it through NN1.

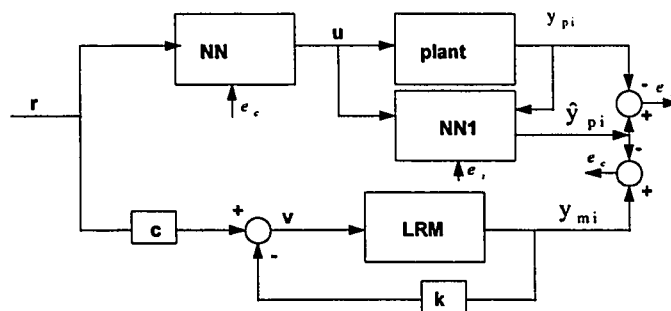


Fig.3.9.An alternative solution using indirect adaptive control method

3.4 Training the Network

The goal of training the network is to adjust the weights so that the desired response is achieved. Usually a neural network is trained over a number of training pairs. Before starting the training process, it is required to initialize all weights to a small random numbers. As mentioned in the previous section, backpropagation starts by presenting an input pattern vector to the input layer of the neural network, propagating forward through the network to generate an output response, and computing the error at each output. Then the errors are propagated backward through the network to associate a square error derivative with each gradient of each derivative. Finally the weights are adjusted based upon the corresponding gradient, producing a new pattern and the process is repeated until the target is achieved. This scheme will not work properly if the initial weights are poorly chosen. The following steps summarize the backpropagation method [36].

1. Present a continuous valued input vector X_0, X_1, \dots, X_{n-1} and specify the target output Y_0, Y_1, \dots, Y_{m-1} . The input could be new on each trial or samples from a training set presented cyclically.
2. Calculate the actual network output.
3. Calculate the error between the desired and actual outputs.
4. Adjust the weights using a recursive algorithm starting at the output layer and working backward to the first hidden layer. Adapt weights using equation 3.13 with the error given in equation 3.22 as the error at the output layer, and equation 3.23 as the error in the hidden layer with the approximation of the unknown partial derivative given by equation 3.21.
5. Repeat step 1 through 4 until the goal is reached.

The backpropagation algorithm is by no means perfect and suffers from the slow convergence problem. In order to speed up the convergence rate, there are many trends in the area of modification of back propagation algorithm. The conjugate gradient method [23], the Newton method, Davidon algorithm [7], the SL-CONE method [29], and the backpropagation algorithm with the momentum term [36], are just a few examples of the proposed methods to speed up the convergence rate of the conventional backpropagation algorithm. In this thesis, the backpropagation algorithm with a momentum term will be used in order to speed up the convergence rate. So, a momentum term of the form $\alpha (w_{ij}^s(k) - w_{ij}^s(k-1))$ will be added to equation 3.16 such that the weights will be adjusted by the following relation [36]:

$$w_{ij}^s(k+1) = w_{ij}^s(k) + \eta e_i^s x_j^{s-1} + \alpha (w_{ij}^s(k) - w_{ij}^s(k-1)) \quad 3.29$$

where $0 < \alpha < 1$.

CHAPTER 4

FEEDBACK LINEARIZATION

The input-state feedback linearization principle is described in this chapter. Feedback linearization is an approach to nonlinear control design. The basic idea of this method is to algebraically transform nonlinear system dynamics into a linear one, so that linear control design techniques can be applied. More precisely, feedback linearization amounts to canceling the nonlinearities in a nonlinear system, so that the closed loop dynamics are in a linear form. This method differs from the other tools in that it is achieved by exact state transformation and feedback.

There are two forms of feedback linearization input-state and input-output linearization. Input-state linearization is achieved by transforming the system's differential equation into the controllability canonical form. Once this is done a linearizing control can easily be designed. The input-output linearization relies on differentiating the output of interest until the input is related to the derivative of the output. This thesis is initially concerned with only the input-state feedback linearization.

A system is said to be in a controllable canonical form if its dynamics are given by [47]:

$$\frac{d^n x}{dt^n} = f(X) + g(X)u \quad 4.1$$

where u is the scalar control input, and $X \in \mathbb{R}^n$ is the state vector:

$$X = \left[x \quad \frac{dx}{dt} \quad \cdot \quad \cdot \quad \frac{d^{n-1}x}{dt^{n-1}} \right]^T \quad 4.2$$

and n is the number of states, and the prime sign stands for transpose. In state space representation this can be written in the following form:

$$\begin{aligned} \frac{dx_1}{dt} &= x_2 \\ &\cdot \\ &\cdot \\ \frac{dx_n}{dt} &= f(X) + g(X)u \end{aligned} \quad 4.3$$

In a matrix representation, this can be written as:

$$\frac{d}{dt} \begin{bmatrix} x_1 \\ \cdot \\ \cdot \\ x_n \end{bmatrix} = \begin{bmatrix} x_2 \\ \cdot \\ \cdot \\ f(X) + g(X)u \end{bmatrix} \quad 4.4$$

In the following section important mathematical tools are introduced.

4.1 Mathematical Tools

The objective of this section is to introduce some mathematical results from differential geometry on which feedback linearization is based. As given in [19] a scalar function $h: \mathbb{R}^n \rightarrow \mathbb{R}$, is said to be smooth if it has continuous partial derivatives of any order. Also, a vector function $f: \mathbb{R}^n \rightarrow \mathbb{R}^n$ defined as a vector field in \mathbb{R}^n is said to be a smooth vector field if it has continuous partial derivatives of any order. The gradient of

a smooth scalar function $h(x)$ is defined as the row vector of dimension n , $\nabla h = \frac{\partial h}{\partial x}$

whose elements are:

$$(\nabla h)_i = \frac{\partial h}{\partial x_i} \quad 4.5$$

Similarly, given a smooth vector field $f(x)$, the Jacobian of f is denoted by ∇f and defined by the $n \times n$ matrix $\nabla f = \frac{\partial f}{\partial x}$ whose elements are:

$$(\nabla f)_{ij} = \frac{\partial f_i}{\partial x_j} \quad 4.6$$

Lie Derivatives And Lie Brackets

For a smooth scalar function h and a smooth vector field f on \mathbb{R}^n , the Lie derivative of h with respect to f is another scalar function defined by [19]:

$$L_f h = \nabla h \cdot f \quad 4.7$$

Which is the directional derivative of h along the direction of the vector field f . Also, a recursive definition can be assigned to higher order Lie derivatives as follows:

$$L_f^0 h = h \quad 4.8$$

$$L_f^i h = L_f(L_f^{i-1} h) \text{ for } i=1,2,\dots,n \quad 4.9$$

Similarly, for a two smooth vector field f and g on \mathbb{R}^n , the Lie bracket of f and g is a third vector field on \mathbb{R}^n and is defined by:

$$\text{ad}_f g = [f, g] = \nabla g \cdot f - \nabla f \cdot g \quad 4.10$$

where ad. stands for adjoint. Also, repeated Lie brackets can be defined recursively by:

$$\text{ad}_f^0 g = g \quad 4.11$$

$$\text{ad}_f^i g = [f, \text{ad}_f^{i-1} g] \quad \text{for } i=1,2,\dots,n \quad 4.12$$

A linearly independent set of vector fields $\{f_1, f_2, \dots, f_m\}$ on \mathbb{R}^n is said to be solvable or completely integrable if and only if there exist $n-m$ scalar functions $h_1(x) \cdots h_{n-m}(x)$ satisfying the following $m(n-m)$ partial differential equations:

$$\nabla h_i f_j = 0 \quad 4.13$$

where $1 \leq i \leq n - m$, and $1 \leq j \leq m$, and n is the dimension of the space and m is the number of vector fields.

Involutivity And Frobenious Theorem

The concept of involutivity will now be defined. As in[19], a linearly independent set of vector fields $\{f_1, f_2, \dots, f_m\}$, is said to be involutive if the Lie bracket of any pairs of vector fields from this set results in a linear combination of the set's vector field. In other words, a linearly independent set of vector fields $\{f_1, f_2, \dots, f_m\}$ is said to be involutive if and only if there are scalar functions $\alpha_{ijk}(x)$ such that

$$[f_i, f_j] = \sum_{k=1}^m \alpha_{ijk}(x) f_k(x) \quad \forall i,j \quad 4.14$$

Another important concept is the Frobenious theorem which provides a necessary and sufficient conditions for the solvability of a special class of partial differential equations. This theorem states that the set of vector fields $\{f_i, f_j\}$ is solvable (completely integrable) if and only if it is involutive. That is to say, complete integrability is equivalent to involutivity.

Diffeomorphism

A diffeomorphism can transform a nonlinear system into another equivalent nonlinear system, but in a simpler form. A nonlinear function $T(\mathbf{X})$ on \mathbb{R}^n , defined in a region Ω where $\Omega \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$, is said to be a local diffeomorphism if and only if the following conditions are satisfied:

- (i) The diffeomorphism $T(\mathbf{X})$ must be smooth, and
- (ii) The Jacobian matrix $\nabla T(\mathbf{X})$ must be nonsingular at a given point in the region Ω .

In the case that $\Omega = \mathbb{R}^n$, then the diffeomorphism is said to be global [19].

Based on the above mathematical preliminaries, the input-state feedback linearization will be discussed in the following section.

4.2 Input-State Feedback Linearization

The input-state feedback linearization is achieved by a combination of a state transformation and an input feedback. It relies on transforming the system's differential equations into the controllable canonical form and then introducing a linearizing feedback. So, a single input nonlinear system in the form of the following differential equations:

$$\dot{\mathbf{X}} = \mathbf{f}(\mathbf{X}) + \mathbf{g}(\mathbf{X})\mathbf{u} \quad 4.15$$

with $\mathbf{X} \in \mathbb{R}^n$ being the state vector given by eqn. (4.2):

$$\mathbf{X} = \left[\begin{array}{cccc} \mathbf{x} & \frac{d\mathbf{x}}{dt} & \cdot & \cdot \\ & & & \frac{d^{n-1}\mathbf{x}}{dt^{n-1}} \end{array} \right]^T$$

and $f(X)$ and $g(X)$ being smooth vector fields on \mathbb{R}^n is said to be input-state linearizable if there exist a region Ω in \mathbb{R}^n , a diffeomorphism $T: \Omega \rightarrow \mathbb{R}^n$, and a nonlinear feedback control law

$$u = \alpha(X) + \beta(X)v \quad 4.16$$

such that the new state vector $Z = T(X)$ and the new input v satisfy a linear time invariant relation

$$\dot{Z} = AZ + bv \quad 4.17$$

where A and b are in the controllable canonical form, that is

$$A = \begin{bmatrix} 0 & 1 & 0 & \dots & \dots & \dots & 0 \\ 0 & 0 & 1 & \dots & \dots & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \dots & \dots & \dots & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, b = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \dots \\ \dots \\ 1 \end{bmatrix} \quad 4.18$$

It was shown in [19], that the nonlinear system given in (4.15) is input-state feedback linearizable if and only if the following conditions hold:

(i) The set of vector fields $g, \text{ad}_f g, \dots, \text{ad}_f^{n-1} g$, are linearly independent in Ω .

Checking this condition amounts to check:

$$\text{rank}[g, \text{ad}_f g, \dots, \text{ad}_f^{n-1} g] = n \quad 4.19$$

This condition simply represents the controllability condition of the nonlinear system in (4.15). The independence of the set of vector fields is equivalent to the invertibility of the linear controllability matrix:

$$\begin{bmatrix} \mathbf{b}, A \mathbf{b}, \dots, A^{(n-1)} \mathbf{b} \end{bmatrix} \quad 4.20$$

(ii) The set of vector fields $[\mathbf{g}, \text{ad}_f \mathbf{g}, \dots, \text{ad}_f^{n-2} \mathbf{g}]$ are involutive in Ω .

If the above two conditions hold, then there exist a nonsingular state transformation $T(\mathbf{X}): \Omega \rightarrow \mathbb{R}^n$ that transforms the nonlinear system into the controllability canonical form, and a nonlinear feedback control law \mathbf{u} which cancels the nonlinear terms in the n -th partial differential equation.

Now, the objective is to investigate the input-state linearization. If there exist a nonsingular state transformation

$$\mathbf{Z} = T(\mathbf{X}) = \begin{bmatrix} T_1(\mathbf{X}) \\ \vdots \\ T_n(\mathbf{X}) \end{bmatrix} \quad 4.21$$

and a nonlinear feedback control

$$\mathbf{u} = \alpha(\mathbf{X}) + \beta(\mathbf{X})\mathbf{v} \quad 4.22$$

such that

$$\dot{\mathbf{Z}} = \frac{\partial T(\mathbf{X})}{\partial \mathbf{X}} \dot{\mathbf{X}} = \frac{\partial T(\mathbf{X})}{\partial \mathbf{X}} (f(\mathbf{X}) + g(\mathbf{X})\mathbf{u}) \Big|_{\mathbf{X}=T^{-1}(\mathbf{Z})} = \mathbf{AZ} + \mathbf{bv} \quad 4.23$$

Expanding the above equation (4.23) yields to [27]:

$$\begin{aligned}
L_f T_1 + L_g T_1 u &= T_2 \\
L_f T_2 + L_g T_2 u &= T_3 \\
&\vdots \\
&\vdots \\
L_f T_{n-1} + L_g T_{n-1} u &= T_n \\
L_f T_n + L_g T_n u &= v
\end{aligned} \tag{4.24}$$

Since $T(X)$ is independent of u , then the above equation (4.24) holds if and only if the following partial differential equations are satisfied:

$$L_g T_1 = L_g T_2 = \dots = L_g T_{n-1} = 0 \tag{4.25}$$

$$L_f T_1 = T_2, \dots, L_f T_{n-1} = T_n \tag{4.26}$$

$$L_g T_n \neq 0 \tag{4.27}$$

which are equivalent to

$$L_g L_f^i T = 0 \quad i = 0, 1, 2, \dots, n-2 \tag{4.28}$$

$$L_g L_f^{n-1} T \neq 0 \tag{4.29}$$

Then, the new state transformation is given by

$$T(X) = \begin{bmatrix} T_1 & L_f T_1 & \dots & L_f^{n-1} T_1 \end{bmatrix}' \tag{4.30}$$

and the nonlinear feedback control is given by :

$$u = \alpha(X) + \beta(X)v \tag{4.31}$$

where v is the new input, and

$$\alpha(X) = \frac{L_f^{n-1} T_1}{L_g L_f^{n-1} T_1} \tag{4.32}$$

and

$$\beta(\mathbf{X}) = \frac{1}{L_g L_f^{n-1} T_1} \quad 4.33$$

Using the state transformation given by eqn.(4.30) and the input feedback given by the relation in (4.31), the nonlinear system in (4.15):

$$\dot{\mathbf{X}} = \mathbf{f}(\mathbf{X}) + \mathbf{g}(\mathbf{X})\mathbf{u}$$

will be transformed into a linear time-invariant of the form given by eqn.(4.17):

$$\dot{\mathbf{Z}} = \mathbf{AZ} + \mathbf{bv}$$

Clearly, the input-state feedback linearization method solves the problem of designing the control input \mathbf{u} in order to cancel the nonlinear terms into three steps. First, the nonlinear dynamics is transformed into the controllable canonical form by finding a state transformation $\mathbf{T}(\mathbf{X})$. Then, a linearizing input is found in order to transform the nonlinear system into a linear one. Finally, standard linear control design technique such as pole-placement is applied.

Based on the above discussion, the input-state feedback linearization technique can be performed through the following steps:

1. Construct the vector fields $\mathbf{g}, \text{ad}_f \mathbf{g}, \dots, \text{ad}_f^{n-1} \mathbf{g}$ for the given system.
2. Check both the controllability, and involutivity conditions.
3. If both conditions are satisfied, solve the equations given by (4.28) and (4.29) to find the first state $T_1(\mathbf{X})$
4. Calculate the new state transformation given by

$$\mathbf{T}(\mathbf{X}) = \begin{bmatrix} T_1 & L_f T_1 & \dots & L_f^{n-1} T_1 \end{bmatrix}'$$

5. Calculate the input transformation required for canceling the nonlinearity given by;

$$\mathbf{u} = \frac{\mathbf{v} - L_f^n T_1}{L_g L_f^{n-1} T_1}$$

6. Use a linear control method such as pole placement to design v (This will be explained in the next chapter).

It is clear that the first essential step in input-state feedback linearization is to find a state transformation. Although, it is not an easy task to find such transformation, there are different methods proposed to solve this problem. In the following section, we will discuss some of them.

4.3 State Transformation

In this section different state transformation methods are introduced for different class of nonlinear systems.

4.3.1 Method 1

This method is proposed by Sommer (1980) [34]. It works for a class of nonlinear systems described by the following dynamics:

$$\dot{X} = f(X) + g(X)u \quad 4.34$$

with $f(X)$ and $g(X)$ being two smooth vector fields on \mathbb{R}^n . The objective is simply to find a state transformation

$$Z = T(X): \mathbb{R}^n \rightarrow \mathbb{R}^n \quad 4.35$$

which transform the nonlinear system given in (4.34) into the global nonlinear controller canonical form [39]:

$$\dot{Z} = \begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \\ \cdot \\ \cdot \\ \dot{z}_n \end{bmatrix} = \begin{bmatrix} z_2 \\ z_3 \\ \cdot \\ \cdot \\ f(z) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \cdot \\ \cdot \\ 1 \end{bmatrix} u \quad 4.36$$

Differentiating eqn. 4.35 with respect to time gives:

$$\dot{Z} = \frac{\partial T(X)}{\partial X} \dot{X} \quad 4.37$$

from eqn. 4.33

$$\dot{Z} = \frac{\partial T(X)}{\partial X} (f(X) + g(X)u) \quad 4.38$$

comparing 4.38 with 4.36 yield:

$$\frac{\partial T_i(X)}{\partial X} (f(X)) = T_{i+1} \quad i = 1, 2, \dots, n-1 \quad 4.39$$

and

$$\frac{\partial T_i(X)}{\partial X} g(X) = 0 \quad i = 1, 2, \dots, n-1 \quad 4.40$$

$$\frac{\partial T_n(X)}{\partial X} g(X) = 1 \quad 4.41$$

eqns. 4.39-4.41 can be written equivalently as:

$$L_f^i T_i = T_{i+1} \quad 4.42$$

$$L_g T_1 = L_g T_2 = \dots = L_g T_{n-1} = 0 \quad 4.43$$

$$L_g T_n = 1 \quad 4.44$$

writing 4.43 and 4.44 in a matrix form gives:

$$\begin{bmatrix} L_g z_1 & \cdot & \cdot & L_g z_{n-1} & L_g z_n \end{bmatrix} = \begin{bmatrix} 0 & \cdot & \cdot & 0 & 1 \end{bmatrix} \quad 4.45$$

Also, from 4.42

$$Z = \begin{bmatrix} z_1 & L_f z_1 & \cdot & \cdot & L_f^{n-1} z_1 \end{bmatrix} \quad 4.46$$

If the matrix on the left hand side of eqn 4.45 is denoted by ζ and is nonsingular, then the last row of ξ^{-1} , represented by ϑ , satisfies the following equation:

$$\vartheta \xi = \begin{bmatrix} 0 & \cdot & \cdot & 0 & 1 \end{bmatrix} \quad 4.47$$

Thus, to solve for T_1 it is required that the following condition is satisfied:

$$\frac{\partial T_1(x)}{\partial x} = \vartheta \quad 4.48$$

If, on the other hand, eqn. 4.48 is not satisfied, then it can be made exact by using an integrating factor.

4.3.2 Method 2

This method works with nonlinear systems in the form given by the following equation [39]:

$$\begin{aligned}\dot{x}_i &= f_i(x_1, x_2, \dots, x_{i+1}) \quad i = 1, 2, \dots, n-1 \\ \dot{x}_n &= f_n(x_1, x_2, \dots, x_n, u)\end{aligned}\quad 4.49$$

using the following state transformation:

$$T(x) = \begin{bmatrix} x_1 \\ f_1(x_1, \dots, x_n) \\ \frac{d}{dt} f_1(x_1, \dots, x_n) \\ \vdots \\ \frac{d^{n-2}}{dt^{n-2}} f_1(x_1, \dots, x_n) \end{bmatrix}\quad 4.50$$

will transform the original system into the controllability canonical form. Once this is done, the linearizing input is easily introduced as mentioned in the previous section.

4.3.3 Method 3

This method proposed in [35] works with nonlinear systems of the form given by 4.35:

$$\dot{X} = f(X) + g(X)u$$

with $f(X)$ and $g(X)$ are two smooth vector fields on R^n . In [35], it was shown that the existence of a local state transformation $T = (T_1, T_2, \dots, T_{n+1})$ under the controllability and the involutivity conditions must satisfy the following partial differential equations:

$$L_g T_1 = L_g T_2 = \dots = L_g T_{n-1} = 0 \quad 4.51$$

$$L_f T_i = T_{i+1} \quad i = 1, 2, \dots, n-1 \quad 4.52$$

$$L_{f+gu} T_n = T_{n+1} \quad 4.53$$

One should note that T_1, T_2, \dots, T_n are functions of x_1, x_2, \dots, x_n only and T_{n+1} is a function of x_1, x_2, \dots, x_n and u , if T_1 is known, then T_2, T_3, \dots, T_{n+1} can be found by solving the following partial differential equations:

$$L_{\text{ad}_f^k} T_1 = 0 \quad k = 0, 1, \dots, n-2 \quad 4.54$$

$$L_{\text{ad}_f^{n-1}} T_1 \neq 0 \quad 4.55$$

In general, it is difficult to solve eqns. 4.54 and 4.55. In [27], a solution to such problem is suggested. It was shown that these equations can be solved by introducing the parameters s, t_1, \dots, t_{n-1} as follows:

First: solve for all $s \in \mathbb{R}$ the system

$$\frac{dx}{ds} = \text{ad}_f^{n-1} g \quad 4.56$$

with initial conditions $x(0) = 0$

Next: solve for all $t_1 \in \mathbb{R}$ the system

$$\frac{dx}{dt_1} = \text{ad}_f^{n-2} g \quad 4.57$$

with initial conditions $x(s, 0) = x(s)$

This procedure is repeated until the last step being to solve for all $t_{n-1} \in \mathbb{R}$ the system

$$\frac{dx}{dt_{n-1}} = g \quad 4.58$$

with initial conditions $x(s, t_1, \dots, t_{n-2}, 0) = x(s, t_1, \dots, t_{n-2})$

The problem of solving the partial differential equations in 4.54 and 4.55 is solved once the parameters s, t_1, \dots, t_{n-1} are found as functions of x_1, x_2, \dots, x_n which depends on the following Jacobian matrix being nonsingular:

$$\begin{bmatrix} \frac{\partial x_1}{\partial s} & \frac{\partial x_1}{\partial t_1} & \cdot & \cdot & \cdot & \frac{\partial x_1}{\partial t_{n-1}} \\ \cdot & \cdot & & & & \cdot \\ \cdot & \cdot & & & & \cdot \\ \cdot & \cdot & & & & \cdot \\ \frac{\partial x_n}{\partial s} & \frac{\partial x_n}{\partial t_1} & \cdot & \cdot & \cdot & \frac{\partial x_n}{\partial t_{n-1}} \end{bmatrix} \quad 4.59$$

The above three methods are equivalent. Also, the existence of such transformation is restricted by satisfying the controllability and involutivity conditions.

4.3.4 Method 4

Although the above mentioned methods are applied to static feedback linearization, a generalized controller canonical form for nonlinear dynamics is introduced briefly in this section. This method is proposed by Fliess in [30]. It works for nonlinear dynamics of the form:

$$\dot{X} = F(X, u) \quad 4.60$$

where F is a function of the components of both X and u . It was shown that there exist a set of state variables $\eta = (\eta_1, \dots, \eta_n)$ such that, in these variables, the equations in (4.60) assume the following generalized controller canonical form:

$$\begin{aligned} \dot{\eta}_1 &= \eta_2 \\ \dot{\eta}_2 &= \eta_3 \\ &\cdot \\ &\cdot \\ \dot{\eta}_{n-1} &= \eta_n \\ \dot{\eta}_n &= c(\eta, u, \dot{u}, \dots, u^{(\alpha)}) \end{aligned} \quad 4.61$$

where the state transformation from X to η involves the control variables and a finite number of their derivatives

$$\eta_i = \tau_i(x, u, \dot{u}, \dots, u^{(\alpha)}) \quad i = 1, 2, \dots, n \quad 4.62$$

where \mathbf{c} and τ_i 's are algebraic functions of their arguments, and may not be defined everywhere, and α is the derivative order.

As mentioned in [30], under such transformation, all nonlinear dynamics can be exactly linearized via a dynamic feedback.

In [2], Levin and Narrendra have shown that there exist a nonlinear transformation that transforms any nonlinear systems into the controllable canonical form provided that they are feedback linearizable.

In this thesis, the proposed scheme is mainly constructed to perform the input-state feedback linearization method based on the idea of the first three state transformation methods. That is to say, no dynamic feedback is used, and this can be considered as a future work to construct the proposed neural network such that any nonlinear system can be linearized via a dynamic feedback.

4.3 Limitations of Feedback Linearization

The feedback linearization based on transforming nonlinear systems into a linear one by using state feedback has been successfully applied to a number of practical control problems. It has been used as a controller design tool. However, this method has a number of important limitations:

1. It is only applied for feedback linearizable nonlinear systems: Systems which satisfy the controllability and involutivity conditions. It cannot be applied to non feedback linearizable systems.
2. It is an off-line method.
3. Input-state feedback linearization is not valid in the whole space, it is only locally feedback linearizable.
4. Exact dynamics of the nonlinear plant must be known.

5. The partial differential equations defining input-state transformation are solved analytically which is not systematic.
6. The full state has to be measured.
7. In the presence of model uncertainties, cancellation of the nonlinearities does no longer take place.

CHAPTER 5

FEEDBACK LINEARIZATION

USING NEURAL NETWORKS

5.1 Motivation

The motivation for the use of neural networks in feedback linearization is provided in this chapter. The proposed diagram schemes are also provided.

A common problem in control is to provide the correct input to drive a possibly nonlinear plant from an initial state to a desired state. The typical approach used in solving such a problem involves linearizing the plant around a number of operating points, introducing linear state-space models of the plant at the operating points, and designing a controller based on the linear control methods. This approach is only valid over a small range of operation. For realistic systems it is usually computationally involved and requires considerable design efforts.

The feedback linearization method is another way of solving such a problem. Although, it is valid over a range greater than that realized by conventional linearization techniques, it is not valid over the whole space. The feedback linearization as mentioned

in chapter 4 is an algebraic off-line technique. The method has,also, a number of important limitations; It is only applicable to a certain class of nonlinear systems which satisfy the controllability and involutivity conditions; Systems which are called feedback linearizable. Nonlinear systems which are non feedback linearizable cannot be handled by this method. In addition, the full states have to be available for measurement. Also, no robustness is guaranteed in the presence of parameter uncertainty or unmodeled dynamics. This means that feedback linearization is sensitive to modeling errors and parameter variations. Besides, the exact dynamics of the nonlinear systems must be precisely known. The method is not systematic, because the nonlinear transformations and the nonlinear feedback need to be solved analytically depending on the exact dynamics of the nonlinear systems.

The objective of this work is to propose a neural network structure that serves as a direct adaptive feedback linearization controller. The proposed architecture is used to remove some of the feedback linearization method drawbacks. It is used to generate the required input signal that drives the nonlinear systems from its initial states to the desired linear states. That is to say, the proposed controller provides the desired linearizing input signal required to transform the nonlinear system into a linear time-invariant system. The proposed controller will perform input-state feedback linearization adaptively and in an on-line manner. Also, it is applied to unknown nonlinear systems using the measured data obtained during the process of the plant, and ,therefore, exact dynamics are no longer needed. In addition, the proposed neural network controller can handle nonlinear systems which are in the controllability canonical form and those which are not in the controllability canonical form. Thus, it avoids the analytical calculations of the state transformations and the nonlinear feedback controller, though this method is systematic. In the following section basic concepts and definitions in control theory are introduced.

5.2 Basic Concepts And Definitions

In this section, many concepts and definitions from system theory perspectives will be discussed.

Definition 1:(Autonomous and Non-Autonomous Systems) Autonomous systems are those which do not depend explicitly on time (i.e. time invariant nonlinear systems). Otherwise, the system is called non-autonomous [49].

Definition 2:(Equilibrium Point) A point $x = x^*$ is an equilibrium point if it has the property that if the state of the system reaches this point, then it will remain there forever [19]. It was shown that an equilibrium point is asymptotically stable, if the Jacobian linearization of the original nonlinear system at the equilibrium point is stable. Also, the equilibrium point is unstable, if the linear approximation of the original nonlinear system is unstable[19].

Definition 3:(Controllability) The controllability of a dynamical system implies the ability to transfer the system from any initial state to a desired state in finite time by using a suitable input signal. From linear control theory, a dynamical system is said to be controllable at t_0 , if there exist a finite time $t_1 > t_0$ such that for any x_0 at t_0 and x_1 in the state space vector, there exists an input $u_{[t_0, t_1]}$, that will transform the state x_0 to the state x_1 at time t_1 . Otherwise, the dynamical system is said to be uncontrollable. This means that the input u is required to be capable of moving any state in the state space to any other state in a finite time; regardless of what trajectory the state should take, and with no constraint imposed on the input (i.e. the magnitude of the input can be as large as desired). For nonlinear systems, the controllability conditions are very hard to establish, therefore, only local controllability conditions are considered. A system is said

to be locally controllable around an equilibrium point $x = x^*$ if for every neighborhood W of x^* , there is a neighborhood Ω of x^* such that for any two states $x_1, x_2 \in \Omega$, there exist an input signal that will transfer the system from x_1 to x_2 in a finite time without leaving W [2,6].

Definition 4:(Observability) a dynamic system is said to be observable at t_0 if there exists a finite $t_1 > t_0$ such that for any state x_0 , the knowledge of the input $u_{[t_0, t_1]}$ and the output $y_{[t_0, t_1]}$ over the time interval $[t_0, t_1]$ suffices to determine the state x_0 . Otherwise, the dynamical system is said to be unobservable at t_0 [6].

5.3 Problem Statement

Consider the following single input nonlinear plants time invariant (autonomous) form

$$\dot{X} = f(X) + g(X)u \quad 5.1$$

with $X \in \mathbb{R}^n$ being the state vector given in (5.2), u is a scalar single input.

$$X = \begin{bmatrix} x & \dot{x} & \cdot & \cdot & x^{(n-1)} \end{bmatrix} \quad 5.2$$

and n is the order of the system (i.e. the number of measurable output states).

Our aim is to determine the desired input signal which stabilizes the system around any desired equilibrium point

In order to solve such problem, a prior information concerning the system in (5.1) is required. These include the following assumptions:

Assumption 1: the nonlinear system is locally controllable.

Assumption 2: the order of the plant is known.

Assumption 3: the states are measurable.

Assumption 4: the functions $f(X)$ and $g(X)$ are smooth vector fields on \mathbb{R}^n , i.e. $f(X)$ and $g(X)$ are continuously differentiable.

The first assumption is due to the complexity of establishing and verifying the conditions of controllability for nonlinear systems. While, the second assumption is required in order to know the number of neurons needed in the input layer of the neural network. The third assumption is made to obtain the data of the unknown nonlinear plants during the control process.

5.4 Stabilization Problem

The problem of controlling a system can be divided into two categories: stabilization and tracking. In stabilization problems, a stabilizer is designed such that the state of the closed loop system will be stabilized around an equilibrium point. In tracking problems, a controller is to be designed such that the system will track a time varying signal. In this thesis, we will restrict our attention to the problem of stabilization.

Consider the nonlinear autonomous system described by the following dynamics:

$$\dot{X} = f(X) + g(X)u \quad 5.3$$

with $f(X)$ and $g(X)$ being two smooth vector fields on \mathbb{R}^n , where $X \in \mathbb{R}^n$ being the state vector:

$$X = \begin{bmatrix} x & \dot{x} & \cdot & \cdot & x^{(n-1)} \end{bmatrix} \quad 5.4$$

if the system

$$\dot{x} = Ax + Bu \quad 5.5$$

is the linearization of the original nonlinear system around an equilibrium point. Linear theory tells us that if the linear approximation of the original nonlinear system is controllable, then there exist a linear feedback law

$$u = -Cx^T \quad 5.6$$

where

$$C = [c_0 \quad c_1 \quad \cdot \quad \cdot \quad c_{n-1}] \quad 5.7$$

that stabilize the linearization system in (5.5). Also the same feedback law will make the original nonlinear system locally stable around that equilibrium point. This method was used in designing controllers to stabilize nonlinear system around unstable equilibrium point. However, this is valid over a small range of operation. In order to increase the range of operation, nonlinear control method must be employed. In the following discussion the stabilization through feedback linearization will be introduced.

Consider the nonlinear system in (5.3). If the controllability and involutivity conditions are satisfied, then the system is locally feedback linearizable and there exist a state transformations

$$Z(x) = [z_1 \quad L_f z_1 \quad \cdot \quad \cdot \quad L_f^{n-1} z_1]' \quad 5.8$$

and an input feedback controller given by

$$u = \alpha(X) + \beta(X)v \quad 5.9$$

that will transform the nonlinear system in (5.3) into the linear time invariant system

$$\dot{Z} = AZ + bv \quad 5.10$$

where A and b are in the controllability canonical form, that is

$$A = \begin{bmatrix} 0 & 1 & \cdot & \cdot & 0 \\ & & \cdot & & \\ & & & \cdot & \\ & & & & \cdot \\ 0 & 0 & \cdot & \cdot & 1 \\ 0 & 0 & \cdot & \cdot & 0 \end{bmatrix}, \text{ and } b = \begin{bmatrix} 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 1 \end{bmatrix} \quad 5.11$$

Thus, the problem of stabilizing the nonlinear system:

$$\dot{X} = f(X) + g(X)u \quad 5.12$$

is transformed into the problem of stabilizing the new linear time invariant system in (5.10) using the new input v .

As mentioned in [19], if the output z_1 is required to track a specified trajectory z_{d1} , then the control law:

$$v = \frac{d^n z_{d1}}{dt^n} - CZ \quad 5.13$$

where $C = [c_0 \quad c_1 \quad \cdot \quad \cdot \quad c_{n-1}]$, $Z = [\tilde{z}_1 \quad \dot{\tilde{z}}_1 \quad \cdot \quad \cdot \quad \tilde{z}_1^{(n-1)}]$

n is the number of states, and the prime stands for transpose, and $\tilde{z}_1 = z_1 - z_{d1}$, leads to the tracking error dynamic

$$\frac{d^n \tilde{z}_1}{dt^n} + c_{n-1} \frac{d^{n-1} \tilde{z}_1}{dt^{n-1}} + \cdots + c_0 \tilde{z}_1 = 0$$

which will be asymptotically stable, if the positive constants c_i are chosen properly such that the polynomial $s^n + c_{n-1}s^{n-1} + \cdots + c_0$ is Hurwitz. A special case of tracking problem occurs when z_{d1} is selected to be constant. This is called the stabilization or regulator problem. In this case, the control law given in 5.13 will be:

$$v = -CZ \quad 5.14$$

where

$$Z = \begin{bmatrix} \bar{z}_1 & \dot{z}_1 & \dots & z_1^{(n-1)} \end{bmatrix}$$

Then, such a control law can place the poles anywhere with c_i chosen properly such that the polynomial $s^n + c_{n-1}s^{n-1} + \dots + c_0$ is Hurwitz (i.e. it has all its roots strictly in the left half plane). Using this linear state feedback control law assures that the linear time invariant system in (5.10) is asymptotically stable around z_{d1} . Hence, the original nonlinear system is also locally asymptotically stable around the same equilibrium point.

5.5 Feedback Linearization Using Neural Network

In this section, the proposed neural network feedback linearization controller is discussed. Two different schemes are proposed to solve the problem of determining the desired input signal to drive a nonlinear plant from its initial states to the desired states of a linear time invariant system.

5.5.1 Scheme 1

As mentioned previously in section 4.2, the basic idea of input-state linearization can be stated as follows; Given a feedback linearizable nonlinear system in the form

$$\dot{X} = f(X) + g(X)u$$

there exist a diffeomorphism that transform the system into

$$\frac{d^n x^*}{dt^n} = f(x^*) + g(x^*)u$$

which is known as the controllable canonical form. A nonlinear feedback control law then obviously exists which transform this system (the system in the controllable canonical form) into a LTI system in the form

$$\dot{Z} = AZ + bv$$

This idea can be materialized by the block diagram of fig. 5.1, in which a neural network is used to provide the control law, while in the same context, the diffeomorphic transformation is realized by another neural network. The LTI system to which the original system is exactly equivalent is introduced in this block diagram as a model-reference system for the original nonlinear system.

We have later discovered that Levin and Narrendra [2] have addressed the same problem and came up with almost the same block diagram. Their block diagram differs from our in two respects. First, Levin and Narrendra used a third neural network for identifying the plant even when the dynamics are known, and second, the algorithm used for the adjustment of the weights in the neural network controller is an advanced version of the traditional backpropagation algorithm which they call the dynamic backpropagation algorithm.

In our block diagram, no identification is performed and a much simpler modified back-propagation algorithm is used for the neural network controller.

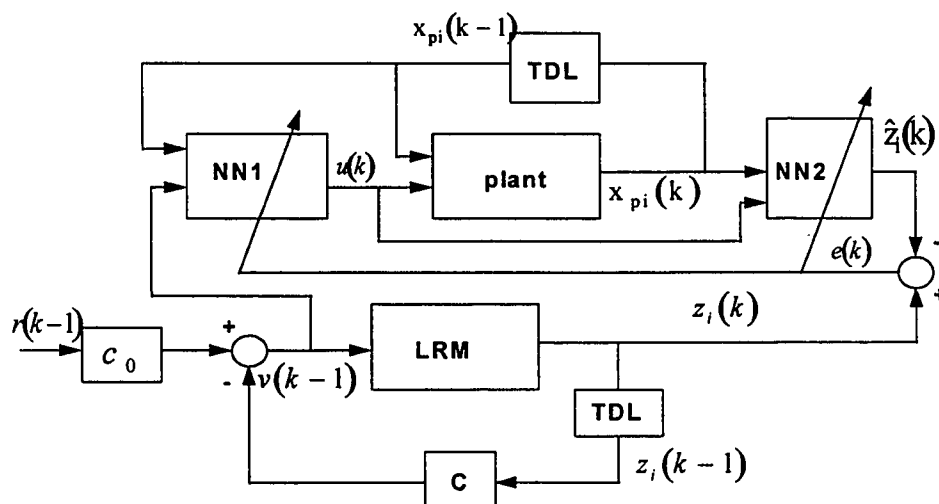


Figure 5.1 Block diagram for scheme 1

The neural network NN1 is used to generate the required linearizing input to drive the nonlinear plant, while NN2 is used to perform state transformations. The output of the neural network NN1 is given by:

$$u(k) = \text{NN1} \left[X_p(k-1), v(k-1) \right] \quad 5.15$$

Where $X_p(k-1) = [x_{p1}(k-1) \ \cdots \ x_{pn}(k-1)]$, and $u(k)$ is the control input at time k , and $x_{pi}(k-1)$ denotes the delayed value of the i th measured output state, v is the new input designed for the control task, and the term TDL stands for time delay.

On the other hand, the output of the second neural network NN2 is given by:

$$\hat{Z}(k) = \text{NN2} \left[X_p(k), u(k) \right] \quad 5.16$$

where $\hat{Z}(k) = [\hat{z}_1(k) \ \cdots \ \hat{z}_n(k)]$, and $X_p(k) = [x_{p1}(k) \ \cdots \ x_{pn}(k)]$ and $\hat{z}_i(k)$ denotes the i -th output of NN2 at time k , and $x_{pi}(k)$ denotes the i -th output of the plant at time k , and $u(k)$ is the output of NN1 which is supposed to be the desired input signal in the plant. Therefore, the output of NN2 can be written as:

$$\hat{Z}(k) = \text{NN2} \left[X_p(k), \text{NN1} \left[X_p(k-1), v(k-1) \right] \right] \quad 5.17$$

Also, note that the output of the plant is given by:

$$X_p(k) = f(X_p(k-1), u(k)) \quad 5.18$$

where the function $f(\cdot)$ is unknown. Thus the plant's output can be written as:

$$X_p(k) = f \left(X_p(k-1), \text{NN1} \left[X_p(k-1), v(k-1) \right] \right) \quad 5.19$$

Note that NN1 and NN2 are two distinct feedforward multilayer networks. In both network, the output layer is chosen to be linear, while the hidden layers are nonlinear tangential sigmoid functions usually taken to be

$$H(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \quad 5.20$$

as discussed in chapter 3.

Both NN1 and NN2 are used to determine the desired input signal and the state transformations adaptively and on-line. Now, the objective is to tune the neural networks NN1 and NN2 simultaneously to produce the desired input signal and the transformations such that the output of the plant $X_p(k)$ and the output of NN2 $\hat{Z}(k)$ will follow the output $Z(k)$ of the linear time invariant system given by (5.10):

$$\dot{Z} = AZ + bv$$

where A, and b are given by (5.11):

$$A = \begin{bmatrix} 0 & 1 & \cdot & \cdot & 0 \\ & & \cdot & & \\ & & \cdot & & \\ & & \cdot & & \\ 0 & 0 & \cdot & \cdot & 1 \\ 0 & 0 & \cdot & \cdot & 0 \end{bmatrix}, \text{ and } b = \begin{bmatrix} 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 1 \end{bmatrix}$$

As mentioned previously, this thesis is concerned only with the problem of stabilization.

Therefore the new input v is selected to be:

$$v(k-1) = -CZ(k-1) + c_0 r(k-1) \quad 5.21$$

where $C = [c_0 \ c_1 \ \cdot \ \cdot \ c_{n-1}]$, and $Z(k-1) = [z_1(k-1) \ \cdot \ \cdot \ z_n(k-1)]^T$, $z_i(k-1)$ is the i -th delayed value of the desired state, and the prime stands for transpose, and r is the reference command signal chosen to be constant signal.

The output error $e(k)$ is given by

$$e(k) = Z(k) - \hat{Z}(k) \quad 5.22$$

and the criterion function to be minimized can be described by

$$E = \frac{1}{2} \sum_i e_i^2(k) \quad 5.23$$

where $e_i(k) = z_i(k) - \hat{z}_i(k)$

and i is the overall neurons in the layer which is taken as the number of the desired output states. If w_2 represent the weights of NN2, then the gradient of E with respect to w_2 is:

$$\frac{\partial E}{\partial w_{2ij}} = -\sum_j (z_i(k) - \hat{z}_i(k)) \frac{\partial \hat{z}}{\partial w_{2ij}} \quad 5.24$$

where w_{2ij} are the weights connecting neuron i to the previous layer's neuron j .

$$\frac{\partial \hat{z}}{\partial w_{2ij}} = \hat{z}^h \quad 5.25$$

where $\hat{Z}^h = [\hat{z}_1^h, \dots, \hat{z}_n^h]$ is the output of the previous layer, and \bar{n} is the number of neurons in that layer (the previous layer).

Since NN2 is directly connected to the output, the weights of NN2 are adjusted using conventional static backpropagation algorithm. However, since the error at the output of NN1 is not available, the output error in (5.22) is propagated backward through the plant. In this case the gradient of E with respect to w_1 , where w_1 represent the weights of NN1 is derived as follows:

$$\frac{\partial E}{\partial w_{1ij}} = -\sum_j (z_i(k) - \hat{z}_i(k)) \frac{\partial x_{pi}}{\partial w_{1ij}} \quad 5.26$$

and

$$\frac{\partial x_{pi}}{\partial w_{1ij}} = \frac{\partial x_{pi}}{\partial u} \frac{\partial u}{\partial w_{1ij}} \quad 5.27$$

and the unknown partial derivative $\frac{\partial x_{pi}}{\partial u}$ is approximated as follows:

$$\frac{\partial x_{pi}}{\partial u} = \frac{x_{pi}(k) - x_{pi}(k-1)}{u(k) - u(k-1)} \quad 5.28$$

Then, the weights w_1 can be adjusted using a static backpropagation algorithm. Through the simulation, we find that the outputs of NN2 which are functions of the nonlinear states values and the input signal u which is the output of NN1 are matching the desired states. Unfortunately, NN1 does not generate the desired linearizing input and we lose the significance of the nonlinear states which are the output of interest. This problem motivates us to modify the neural network structure shown in fig. 5.1, and we end up with the following scheme explained in the following subsection.

5.5.2 Scheme 2

The modified proposed diagram scheme is shown in fig. 5.2 . Clearly, the neural network structure is simpler in architecture than scheme 1. It is clear from the figure that the neural network NN2 in fig. 5.1 which is supposed to handle the state transformation is removed, and we are left only with one neural network NN. This neural network serves as a direct adaptive feedback linearization controller, which will be proved through simulations.

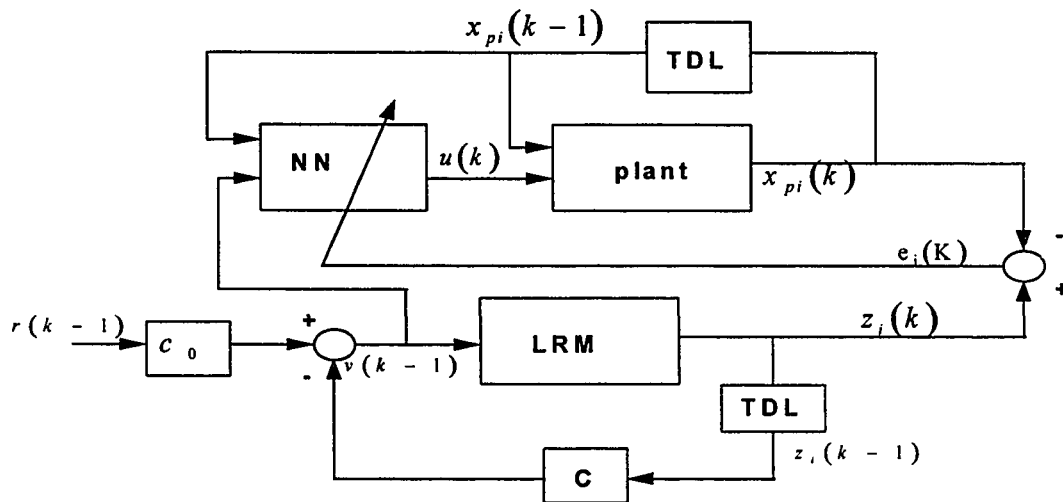


Figure 5.2. Block diagram of feedback linearization controller using NN

The output of the neural network is given by:

$$u(k) = \text{NN} \left[X_p(k-1), v(k-1) \right] \quad 5.29$$

where $X_p(k-1) = [x_{p1}(k-1) \ \cdots \ x_{pn}(k-1)]$, $x_i(k-1)$, and v are the delayed values of the measured output states and the new input to be designed respectively, r is the reference command signal and e is the output error. NN is a feedforward multilayer neural network, with its output layer chosen to be linear, and the output of neurons in the hidden layers are taken to be tangential sigmoid nonlinear function. A tangential sigmoid function is given by

$$H(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \quad 5.30$$

the term TDL stands for time delay.

The output of the plant is given by

$$X_p(k) = f(X_p(k-1), u(k)) \quad 5.31$$

where $f(\cdot)$ is unknown function. Clearly, the plant's output is a function of the delayed values of its output and the output of NN. Thus,

$$X_p(k) = f \left(X_p(k-1), \text{NN} \left[X_p(k-1), v(k-1) \right] \right) \quad 5.32$$

Now, the objective is to tune the neural network NN such that its output is the required linearizing input to force the output of the plant $X_p(k)$ to match the output $Z(k)$ of the linear time invariant system given by (5.10):

$$\dot{Z} = AZ + bv$$

where $X_p(k) = [x_{p1}(k) \ \cdots \ x_{pn}(k)]$, and $Z = [z_1 \ z_2 \ \cdots \ z_n]$, A and b are given by (5.11):

$$A = \begin{bmatrix} 0 & 1 & \cdot & \cdot & 0 \\ & & \cdot & & \\ & & \cdot & & \\ & & \cdot & & \\ 0 & 0 & \cdot & \cdot & 1 \\ 0 & 0 & \cdot & \cdot & 0 \end{bmatrix}, \text{ and } b = \begin{bmatrix} 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 1 \end{bmatrix}$$

and v is chosen to be

$$v(k-1) = -CZ(k-1) + c_0 r(k-1) \quad 5.33$$

where $Z(k-1) = [z_1(k-1) \ \cdot \ \cdot \ z_n(k-1)]$, $C = [c_0 \ c_1 \ \cdot \ \cdot \ c_{n-1}]$ is the gain constant chosen properly such that the polynomial $s^n + c_{n-1}s^{n-1} + \dots + c_0$ is Hurwitz, and $z_i(k-1)$ is the i -th delayed value of the desired state, r is a constant reference command signal.

The output error is given by

$$e(k) = Z(k) - X_p(k) \quad 5.34$$

In this case, the criterion function to be minimized E is derived as follows:

$$E = \frac{1}{2} \sum_i e_i^2(k) \quad 5.35$$

where i is the number of the output states. If w represent the weights of the neural network NN, then the gradient of E with respect to w is derived as follows

$$\frac{\partial E}{\partial w_{ij}} = - \sum_j (z_i(k) - x_{pi}(k)) \frac{\partial x_{pi}}{\partial w_{ij}} \quad 5.36$$

and

$$\frac{\partial x_{pi}}{\partial w_{ij}} = \frac{\partial x_{pi}}{\partial u} \frac{\partial u}{\partial w_{ij}} \quad 5.37$$

The unknown partial derivative $\frac{\partial x_{pi}}{\partial u}$ is approximated as explained in chapter 3 to be

$$\frac{\partial x_{pi}}{\partial u} = \frac{x_{pi}(k) - x_{pi}(k-1)}{u(k) - u(k-1)} \quad 5.38$$

Using the backpropagation algorithm explained in chapter 3, the weights of the neural network NN are adjusted at each instant of time.

The proposed scheme as mentioned above serves as a direct adaptive feedback linearization controller. This idea is original and never found in published papers. There are many advantages of the proposed neural network feedback linearization controller. These can be summarized as follows:

1. Easier to implement.
2. Minimize our efforts. That is because it avoids training stage, and it serves as on-line direct adaptive controller.
3. It avoids state transformation. and perform well and generates the required linearizing input and force the nonlinear states to follow the desired states.
4. Simple architecture
5. Removes the feedback linearization method drawbacks.

CHAPTER 6

SIMULATION RESULTS

The performance of the proposed neural network controller will be investigated and an extensive simulation study will be performed in this chapter. The initial values of the neural network weights are all set to small random quantities. This is so to assure the learning of the network, and to prevent the weights from getting stuck at high values.

It is assumed that the number of states is known. This is required to determine the number of the input neurons in the neural network. Also, it is assumed that the states are measurable.

Through simulation study, we found that a three layered neural networks using the tangential sigmoid function are adequate to generate the required input signal to drive the plant from its initial states to the desired states of a linear time-invariant system. Therefore, in all our simulation, we have fixed the number of layers to 3, with two hidden layers and one output layer. The number of neurons in the input layer is fixed to the total number of the plant's states plus the input v . The number of neurons in the

output layer is fixed to one. But, the number of neurons in the hidden layers is a matter of guesswork and has to be found using simulations. This is because there are no rules which tell the number of neurons in the hidden layer for specific situations.

Through all the simulation results the sampling time is fixed to 10^{-3} second, and x_m in the figures denotes the desired states of the linear time-invariant system.

In the following sections the performance of the proposed neural network controller applied to three different examples is discussed.

6.1 Antenna Arm Control System

The antenna arm control system is one of many interesting problems in stabilization control system. In this section the antenna arm system shown in fig.(6.1) will be controlled using the proposed technique:

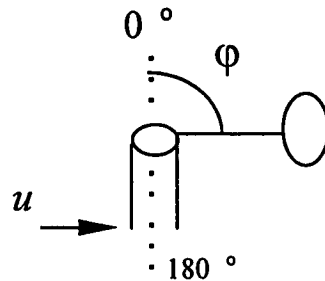


Figure 6.1 Antenna arm control system

This system can be described by the following second order differential equation:

$$\frac{d^2\phi}{dt^2} + 2\frac{d\phi}{dt} - 10\sin\phi = u \quad 6.1$$

where φ is the antenna arm angle to be controlled, the term $2 \frac{d\varphi}{dt}$ represent the viscous friction acting against the velocity , the term $10 \sin \varphi$ is the force of the gravity, and u is a current applied to a DC motor attached to the antenna arm.

In state space representation, the above dynamic can be written as:

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= 10 \sin x_1 - 2x_2 + u\end{aligned}\tag{6.2}$$

where $x_1 = \varphi$ and $x_2 = \dot{\varphi}$.

Setting $\dot{x}_1 = \dot{x}_2 = 0$ and solving for the equilibrium points, it can be verified that equilibrium points have the form

$$X = \left(\sin^{-1}\left(\frac{-u}{10}\right), 0 \right), \text{ where } X = [x_1, x_2]$$

Our goal is to stabilize the antenna arm angle at the horizontal position, that means to control the antenna arm angle at $x_1 = \varphi = 90^\circ$.

In our investigation of the antenna arm system we will limit our attention to the equilibrium point $(90^\circ, 0)$. Physically, we can see that the system can hardly maintain rest at this equilibrium point because the force of gravity will tend to pull the system downward to the ground. So, a sufficient current is needed to maintain the system at this position. Theoretically, this is due to the stability property of the equilibrium point. Lyapunov first method tells us that if the linearized system at the desired equilibrium point is strictly stable, then the equilibrium point is asymptotically stable for the original nonlinear system. Furthermore, if the linearized system is unstable, then the equilibrium point is unstable for the nonlinear system.

The linearized system has the form;

$$\dot{X} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 10 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -10 \end{bmatrix} u = AX + bu\tag{6.3}$$

Checking the eigenvalues of the matrix A , we found that the linearized system is unstable. Therefore, the equilibrium point is unstable. The state space equations in (6.2) can be written in the following form:

$$\dot{X} = f(X) + g(X)u \quad 6.4$$

where $X = [x_1 \quad x_2]$

and
$$f(X) = \begin{bmatrix} x_2 \\ 10 \sin x_1 - 2x_2 \end{bmatrix}, \text{ and } g(X) = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad 6.5$$

It is clear that the given system is in the controllability canonical form. If the linearizing input is chosen as

$$u = v - 10 \sin x_1 + 2x_2 \quad 6.6$$

then, the original nonlinear system will be transformed into the following linear one:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= v \end{aligned} \quad 6.7$$

where v is the new input to be designed for the control task. Suppose that we want the eigenvalues of the linearized closed-loop system to be assigned at $\lambda_{1,2} = -3$. Then, the characteristic equation of the linearized closed loop system has the form $(s+3)(s+3)$.

The state feedback assigning the desired eigenvalues to the linearized closed-loop system is

$$v = -c_0 x - c_1 \dot{x} - \dots - c_{n-1} x^{(n-1)} \quad 6.8$$

where x is the scalar output of interest, in this example $x = x_1 - x_d$, where x_d is the desired output angle, with c_i 's chosen properly such that the polynomial $s^2 + c_1s + c_0$ is Hurwitz (i.e. it has all its roots in the left half plane). Then, the new input leads to the exponentially stable dynamics

$$\ddot{x} + c_1\dot{x} + c_0 = 0 \quad 6.9$$

Thus, if we choose

$$(s + 3)(s + 3) \cong s^2 + c_1s + c_0 \quad 6.10$$

Then, the control law will be

$$\begin{aligned} v &= -9(x_1 - x_d) - 6(\dot{x}_1 - \dot{x}_d) \\ v &= -9(x_1 - x_d) - 6\dot{x}_1 \\ v &= -9x_1 - 6x_2 + 9x_d \end{aligned} \quad 6.11$$

and the linear time-invariant system in (6.7) became

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -9x_1 - 6x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 9 \end{bmatrix} r \quad 6.12$$

where $r = x_d = 90^\circ$.

Using the feedback linearization method the linearizing input became;

$$u = -9x_1 - 6x_2 + 9r - 10 \sin x_1 + 2x_2 \quad 6.13$$

This scheme is, however, an off-line method which requires an interruption of the process in order to tune the gains of the controller, and this is impractical because the dynamics may change during the time when the gains are being updated. Also, this technique requires a precise knowledge of the plant's dynamic.

In contrast, applying the proposed neural network controller will solve this problem, and the dynamics of the plant need no longer to be known. With the proposed neural network controller, we found through the simulation that a neural network with the configuration given in table(1) is adequate to generate the desired input signal to drive the plant from its initial states to the desired states of the linear system in (6.12).

#neurons in input layer	#neurons in 1st hid.layer	#neurons in 2nd hid layer	#neurons in output layer
4	10	5	1

TABLE 1

This problem was contributed by Martin Hagan in [47], where a neural network is used to model the second component of $f(X)$ given in eqn. (6.5) in order to use it to cancel the nonlinearity. That is, the nonlinear term $10 \sin x_1 - 2x_2$ which is precisely known is approximated in an off-line manner by a neural network N_f , after that the linearizing input is calculated as

$$u = v - N_f$$

thus canceling the nonlinearity. The simulation result obtained in [47] is shown in fig. 6.2 which compare the response of the nonlinear system with the neural network controller to that with the perfect cancellation controller (i.e. the linearizing input calculated by the feedback linearization method). It is clear that the neural network does very well compared to the perfect cancellation controller. Clearly, with an initial condition of 10° , the settling time is about 2.5 seconds with no overshoot.

Unfortunately, this method is only applied to nonlinear systems which are in the controllable canonical form.

On the other hand, in our proposed scheme a neural network is used as a direct controller with no training stage using a modified backpropagation algorithm called the backpropagation through the plant algorithm. Also the proposed scheme is performed in an on-line manner for unknown nonlinear dynamics. The simulation results are shown in figures 6.3(a-f) with initial condition $x = \begin{bmatrix} -4 & 5^\circ & 0 \end{bmatrix}$. As shown in the results, the neural network controller generates the required input signal and the nonlinear states followed the desired states after 7000 iterations, that means the settling time is about 7 seconds. Also, a maximum overshoot of +5 radian occurs. As a result, the proposed scheme in [47] is better than our scheme, in the sense that it has a faster response with no overshoot. This perhaps due to the following reasons: The approach proposed in [47] is an off-line scheme with the exact dynamics precisely known. The slow convergence property associated with the back-propagation algorithm which become more severe in an on-line scheme. Starting with different initial condition. Also, different neural network typology is used, in our proposed scheme a 3-layered neural network is used, while 2-layered neural network is used in [47] which is easier to implement.

However, the proposed scheme has the following advantages over the scheme proposed in [47]: It is applicable to unknown nonlinear systems not necessarily in the controllable canonical form. Moreover, it avoids the training stage. Also, it serves as a direct adaptive controller which produces a smooth input to drive the plant to the desired performance as shown in fig. 6.3e.

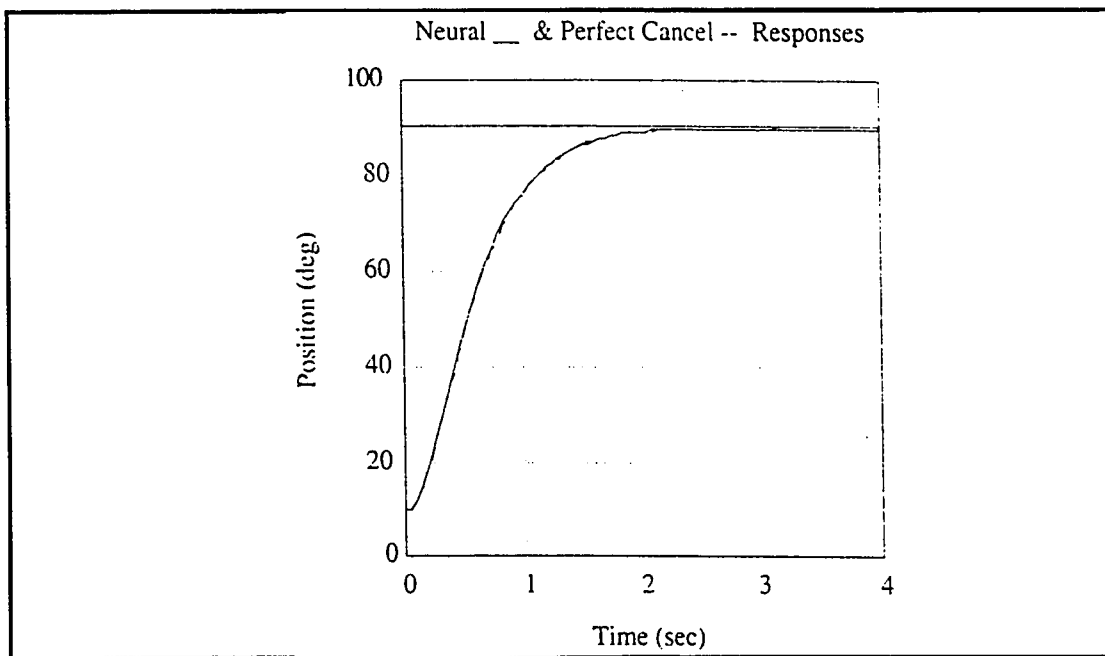


fig. 6.2 The neural network controller response and the perfect cancellation controller response as given in [47].

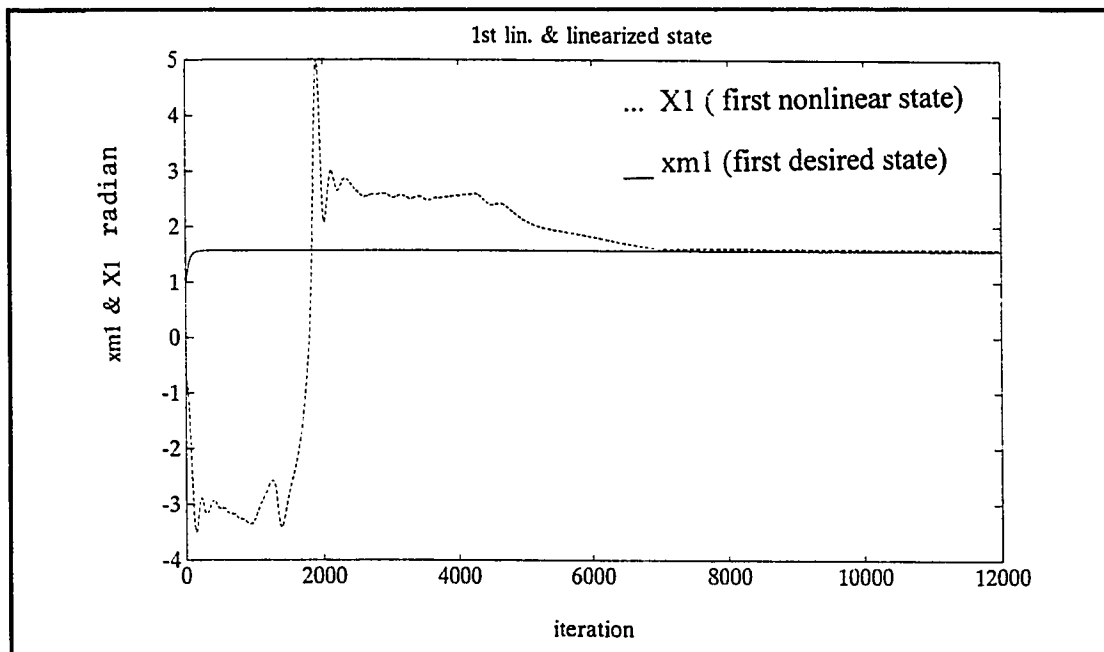


Figure. 6.3a The first linear and linearized state with initial condition; $x_1 = -45^\circ, x_2 = 0$

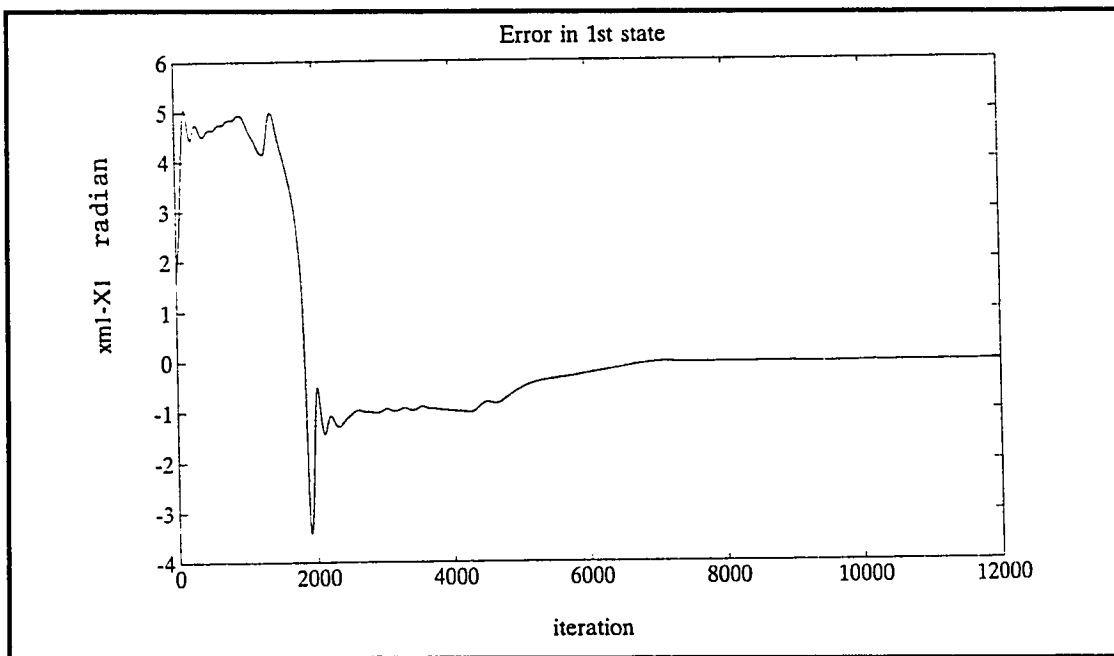


Figure 6.3b The error between the first linear and linearized state

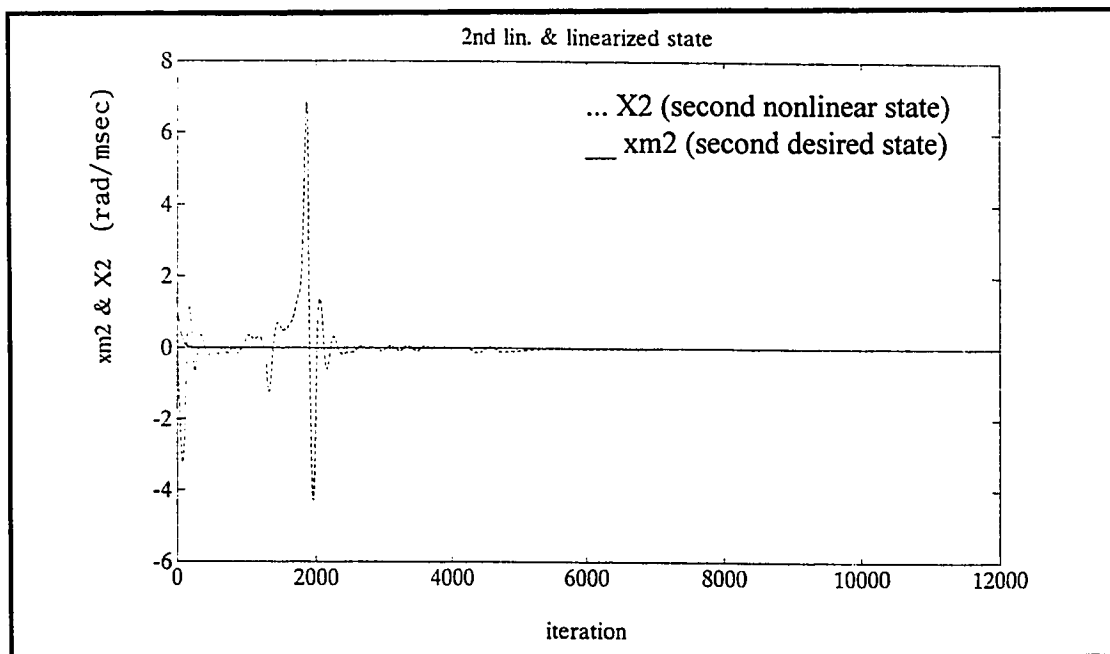


Figure 6.3c The 2nd linear and linearized state with initial condition; $x_1 = -45^\circ, x_2 = 0$.

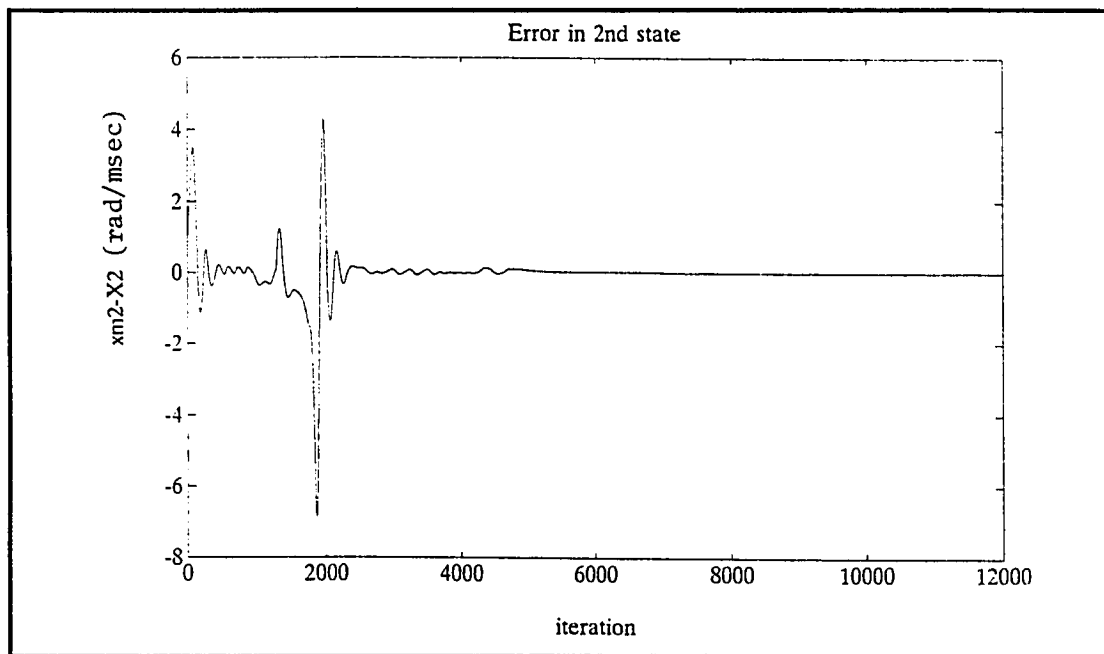


Figure 6.3d The error between the second linear and linearized state

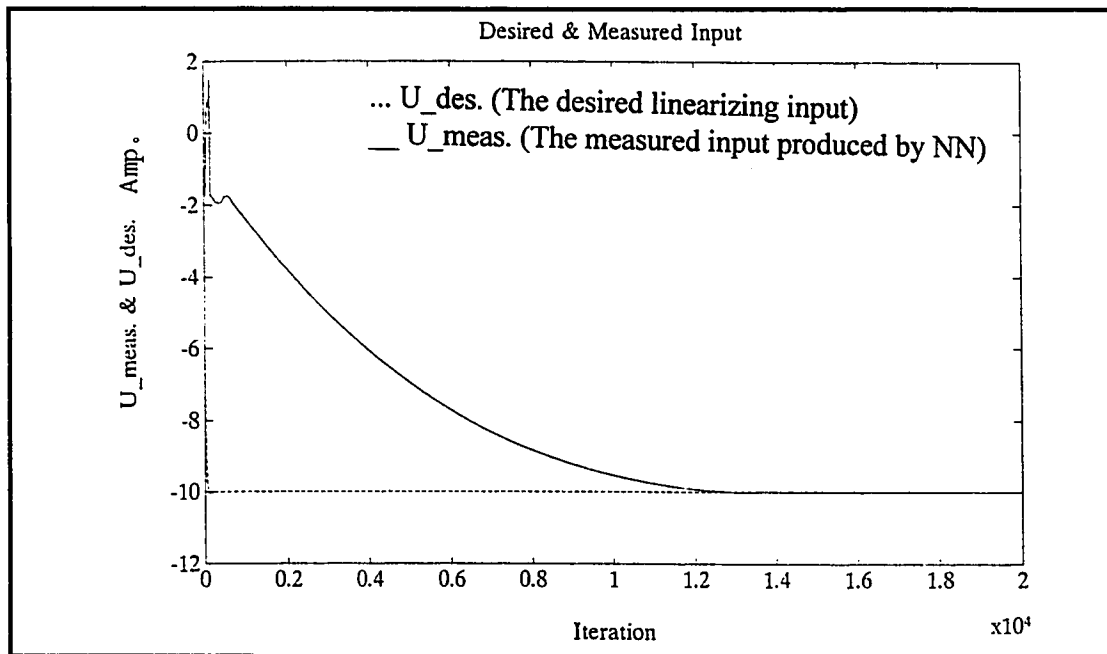


Figure 6.3e The desired linearizing input of the feedback linearization method and the linearizing input produced by the proposed NN controller.

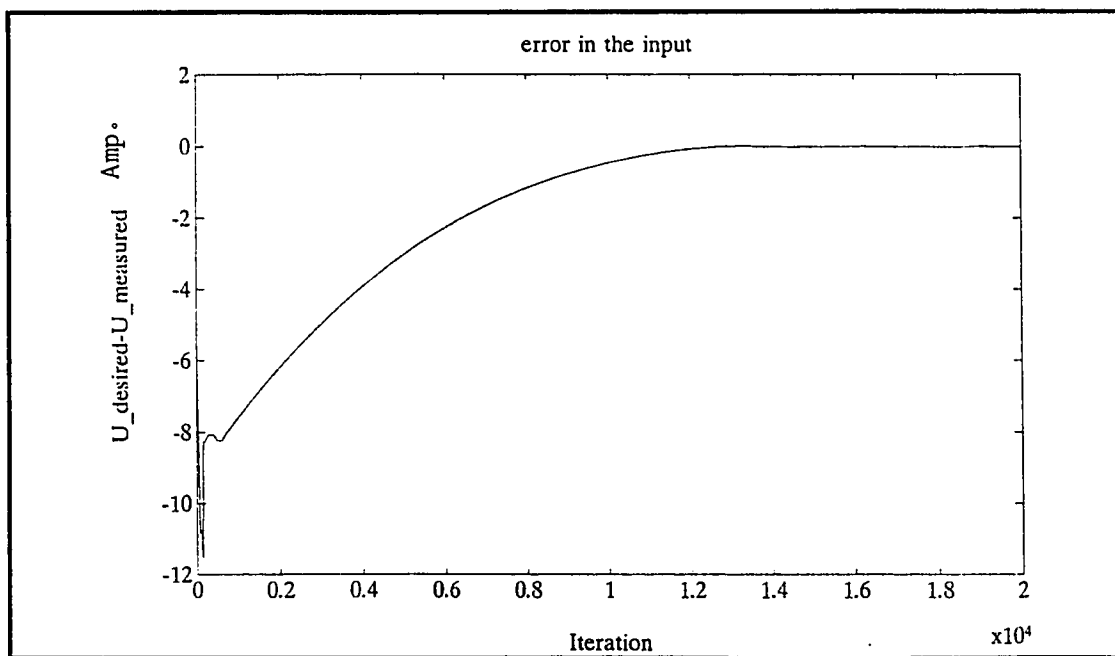


Figure 6.3f The error between the desired linearizing input calculated by feedback linearization and the measured linearizing input produced by NN.

6.2 The Inverted Pendulum Stabilization Problem

There are many examples of unstable systems, one of these examples is the inverted pendulum. It is unstable because the centroid of the system is placed well above the point of suspension. So it cannot remain in the desired vertical position unless there is a control force applied to the base. In this section the proposed controller will be used to stabilize the inverted pendulum at the vertical position.

Let us consider an inverted pendulum with a DC motor control as shown in fig. (6.4). This problem is introduced by [Zak and MacCarley, 1986] [39]. It is assumed that the DC motor is armature controlled, and the motor inertia is negligible when compared with the pendulum inertia. The DC motor is modeled as shown in fig.(6.5).

As given by[39], this system can be described by a third order system:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} x_2 \\ \frac{g}{l} \sin x_1 + \frac{10k_m}{l^2 m} x_3 \\ -\frac{10k_b}{L} x_2 - \frac{R}{L} x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{1}{L} \end{bmatrix} u \quad 6.14$$

Where $x_1 = \phi$ is the angular position, $x_2 = \dot{\phi}$ is the angular velocity, and $x_3 = I$ is the armature DC current, and u is the voltage supplied to the DC motor, and g, l, m, k_m, k_b, R, L are some non zero constants. Reasonable parameters describing our system are

$$g = 9.81 \text{ m/s}^2, l = 1 \text{ m}, k_m = 0.1 \text{ Nm/A}, k_b = 0.1 \text{ Vs/rad}, \\ m = 10 \text{ kg}, R = 1 \Omega, L = 100 \text{ mH}$$

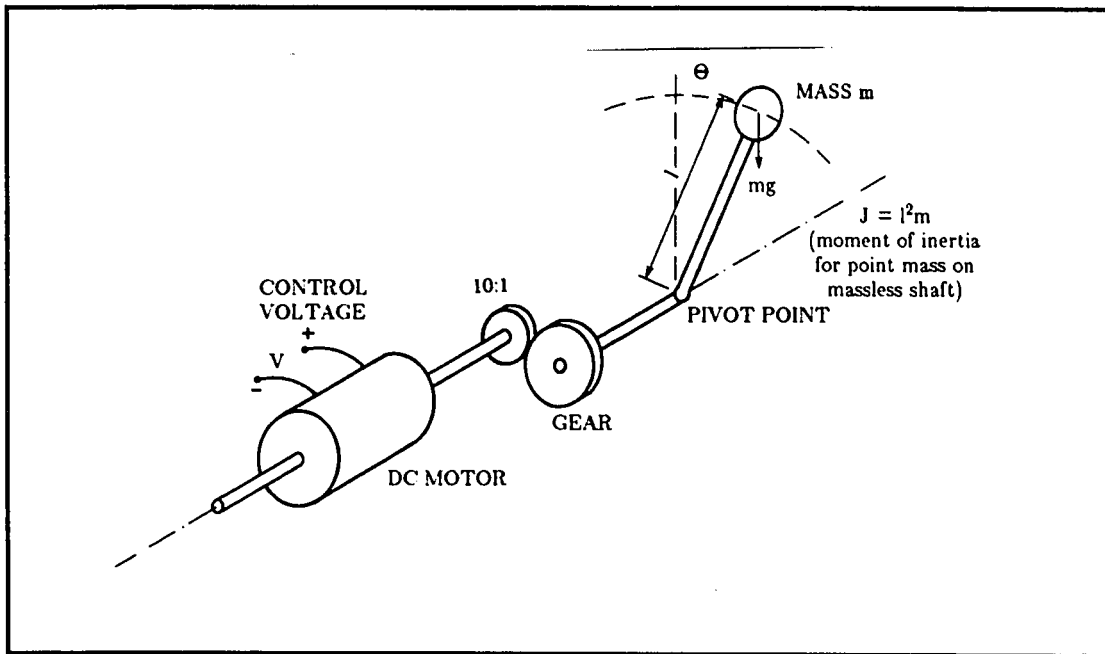


Figure 6.4 Inverted pendulum controlled by a DC motor

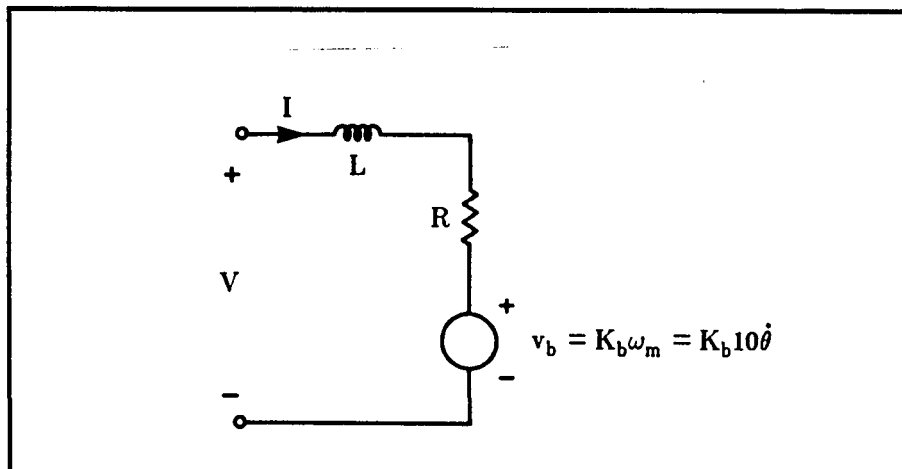


Figure 6.5 Model of an armature-controlled DC motor

In order to simplify the notation, let

$$k_1 = \frac{g}{l}, k_2 = \frac{10k_m}{l^2 m}, k_3 = -\frac{10k_b}{L}, k_4 = -\frac{R}{L}, \text{ and } k_5 = \frac{1}{L}$$

then, the system equation take the form given by eqn. 6.15:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} x_2 \\ k_1 \sin x_1 + k_2 x_3 \\ k_3 x_2 + k_4 x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ k_5 \end{bmatrix} u \quad 6.15$$

Setting $\dot{x}_1 = \dot{x}_2 = \dot{x}_3 = 0$ and solving for the equilibrium points, it can be verified that the set of operating point is;

$$X = \left(\sin^{-1} \left(\frac{k_2 k_5}{k_1 k_4} u \right), 0, -\frac{k_5}{k_4} u \right), \text{ where } X = [x_1, x_2, x_3]$$

In our investigation of the inverted pendulum, we will limit our attention to the zero equilibrium point which corresponds to the vertical position. The linearized model at this equilibrium point has the following form;

$$\dot{X} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ k_1 & 0 & k_2 \\ 0 & k_3 & k_4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ k_5 \end{bmatrix} u = AX + bu \quad 6.16$$

Checking the eigenvalues of the matrix A in (6.14), we found that the linearized system is unstable. Therefore, the equilibrium point is unstable. The parameters entering in eqn. 6.16 are

$$k_1 = 9.8, k_2 = 1, k_3 = -10, k_4 = -10, k_5 = 10$$

The response of the uncontrolled inverted pendulum is shows in figure 6.6. It is clear from the figure that the system maintains rest at the stable equilibrium point $x_1 = \pm n \cdot \pi$, where $\pi = 3.14$ and $n=1,3,5,\dots$

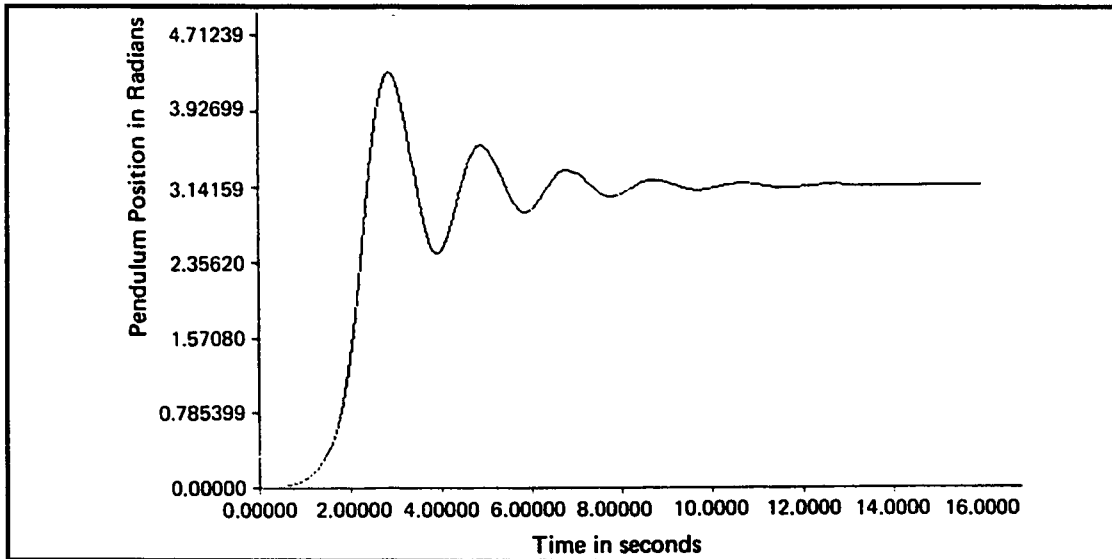


Figure 6.6 The response of uncontrolled inverted pendulum, $u=0$

The nonlinear system in (6.15) can be written in the following form;

$$\dot{X} = f(X) + g(X)u \quad 6.17$$

where $X = [x_1 \quad x_2 \quad x_3]$

and

$$f(X) = \begin{bmatrix} x_2 \\ k_1 \sin x_1 + k_2 x_3 \\ k_3 x_2 + k_4 x_3 \end{bmatrix}$$

and

$$g(X) = \begin{bmatrix} 0 \\ 0 \\ k_5 \end{bmatrix}$$

It is clear that the given system is not in the controllability canonical form, and applying the feedback linearization method requires a state transformation to be found

so that the system is transformed into the controllability canonical form, then a linearizing input is introduced.

Simple calculations give:

$$\text{ad}_f g = \begin{bmatrix} 0 \\ -k_2 k_5 \\ -k_4 k_5 \end{bmatrix}, \text{ and } \text{ad}_f^2 g = \begin{bmatrix} k_2 k_5 \\ k_2 k_4 k_5 \\ k_2 k_3 k_5 + k_4^2 k_5 \end{bmatrix} \quad 6.18$$

and

$$\text{rank} \begin{bmatrix} g & \text{ad}_f g & \text{ad}_f^2 g \end{bmatrix} = 3 \quad 6.19$$

also, the set $\{g, \text{ad}_f g\}$ is involutive, because the vector fields in this set are constant.

As shown in[39], choosing the first state transformation as;

$$T_1 = \frac{1}{k_2 k_5} x_1 \quad 6.20$$

then,
$$T_2 = \frac{1}{k_2 k_5} x_2 \quad 6.21$$

and
$$T_3 = \frac{1}{k_2 k_5} (k_1 \sin x_1 + k_2 x_3) \quad 6.22$$

Thus, the new state transformation is

$$z(x) = \begin{bmatrix} T_1(x) \\ T_2(x) \\ T_3(x) \end{bmatrix} \quad 6.23$$

In the new state transformation the system is expressed as

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \\ \dot{z}_3 \end{bmatrix} = \begin{bmatrix} z_2 \\ z_3 \\ (k_1 \cos(k_2 k_5 z_1) + k_2 k_3) z_2 - k_4 z_3 + \frac{k_1 k_4}{k_2 k_5} \sin(k_2 k_5 z_1) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u \quad 6.24$$

Choosing the linearizing input as:

$$u = v - \left((k_1 \cos(k_2 k_5 z_1) + k_2 k_3) z_2 - k_4 z_3 + \frac{k_1 k_4}{k_2 k_5} \sin(k_2 k_5 z_1) \right) \quad 6.25$$

will transform the original nonlinear system into the following linear form:

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \\ \dot{z}_3 \end{bmatrix} = \begin{bmatrix} z_2 \\ z_3 \\ v \end{bmatrix} \quad 6.26$$

and, the new input v is designed as follows:

$$v = -c_0 \tilde{z} - c_1 \dot{\tilde{z}} - c_2 \ddot{\tilde{z}} \quad 6.27$$

where $\tilde{z} = z_1 - z_d$

with the positive constants c_i 's chosen properly such that the following polynomial

$$s^3 + c_2 s^2 + c_1 s + c_0 \quad 6.28$$

has all its roots strictly in the left half plane, leads to the exponentially stable dynamics

$$s^3 + c_2 s^2 + c_1 s + c_0 \quad 6.29$$

Now, suppose that we want the eigenvalues of the closed-loop linearized system to be assigned at $\lambda_{1,2,3} = -2, -2, -10$. Then, the characteristic equation of the linearized

closed-loop system has the form

$$\begin{aligned} (s+2)^2 (s+10) &\cong s^3 + c_2 s^2 + c_1 s + c_0 & 6.30 \\ s^3 + 14s^2 + 44s + 40 &\cong s^3 + c_2 s^2 + c_1 s + c_0 \end{aligned}$$

Therefore, the control law is

$$v = -40(z_1 - z_d) - 44(\dot{z}_1 - \dot{z}_d) - 14(\ddot{z}_1 - \ddot{z}_d) \quad 6.31$$

where z_d is the vertical position of the inverted pendulum, $z_d = 0^\circ$. Thus the control law

$$v = -40z_1 - 44z_2 - 14z_3 + 40z_d \quad 6.32$$

and the linear time-invariant system in (6.26) became;

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \\ \dot{z}_3 \end{bmatrix} = \begin{bmatrix} z_2 \\ z_3 \\ -40z_1 - 44z_2 - 14z_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 40 \end{bmatrix} r \quad 6.33$$

Where r is the desired vertical position, in this case $r = z_d = 0^\circ$.

Applying the proposed neural network controller, on the other hand, avoids the state transformation and produce the desired input signal to the plant to cancel the nonlinearities and drives the nonlinear states to the desired states of the linear time-invariant system in (6.33).

The following figures contain the simulation results for the proposed neural network controller. In fig. 6.7(a-f) the responses of the closed loop system when subjected to non zero initial condition is depicted. Figure 6.7(g-h) shows a comparison between the linearizing input calculated by feedback linearization method and the linearizing input generated by the proposed neural network control.

Observe that the proposed neural network control with the configuration given in table 2 is capable to drive the nonlinear states to the desired states of the linear time-invariant system.

#neurons in input layer	#neurons in 1st hid.layer	#neurons in 2nd hid. layer	#neurons in output layer
5	10	5	1

TABLE 2

As shown in the figures, the proposed neural controller generates the desired input signal and the nonlinear states agree with the desired states of the linear time-invariant system after 5000 iterations.

Although the stabilization of the inverted pendulum at the vertical position is a very hard problem, it is clear from fig.6.7(a-f) that the proposed neural network which serves as a direct feedback linearization controller stabilized the inverted pendulum at the desired vertical position. further more, it is not only producing the desired linearizing input, but also it handle the problem of the state transformation. This can be considered as an advantage of the proposed neural network control.

In a first stage of this work, we tried to stabilize the inverted pendulum through the structure given in fig. (5.1) which is similar to the block diagram proposed by Levin and Narrendra in [2]. In [2], the training was achieved after 50000 steps, while the neural network structure shown in fig. (5.1) fails to achieve the goal. On the other hand, the proposed scheme shown in fig. (5.2) proves its capability to stabilize the inverted pendulum and forcing the nonlinear states to follow the desired states of the linear time-invariant system after 5000 steps,(i.e. the proposed scheme is simpler and achieves the goal with a steps reduced by a factor of 1/10 when compared to the neural network approach proposed by Levin and Narrendra).

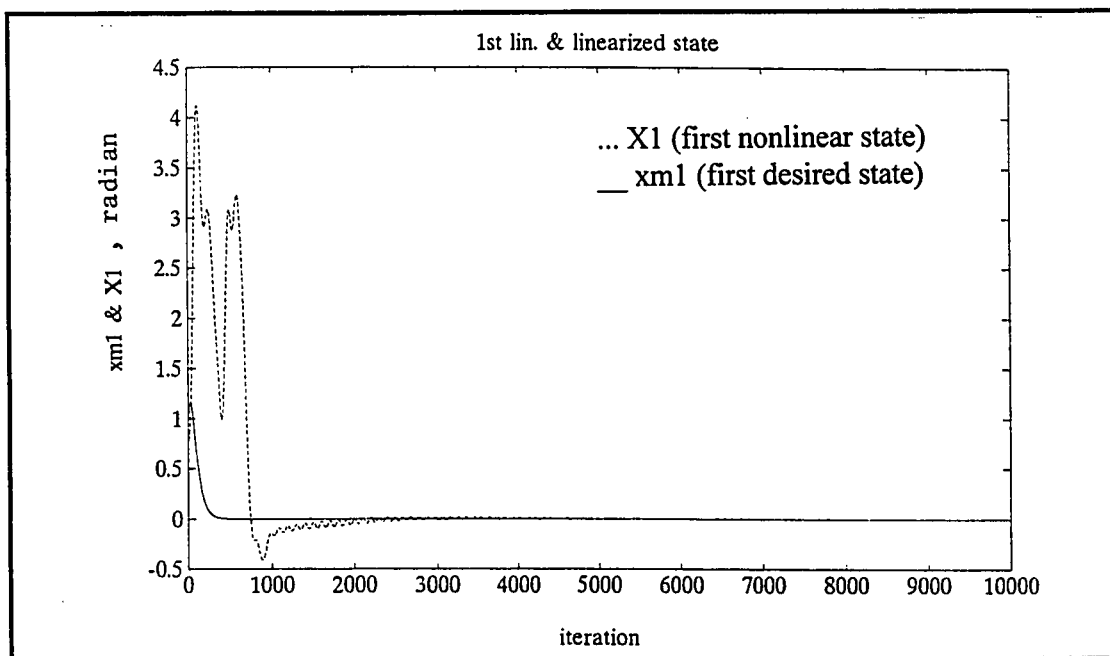


Figure 6.7a The first linear and linearized state with initial condition; $x_1 = 45^\circ, x_2 = x_3 = 0$.

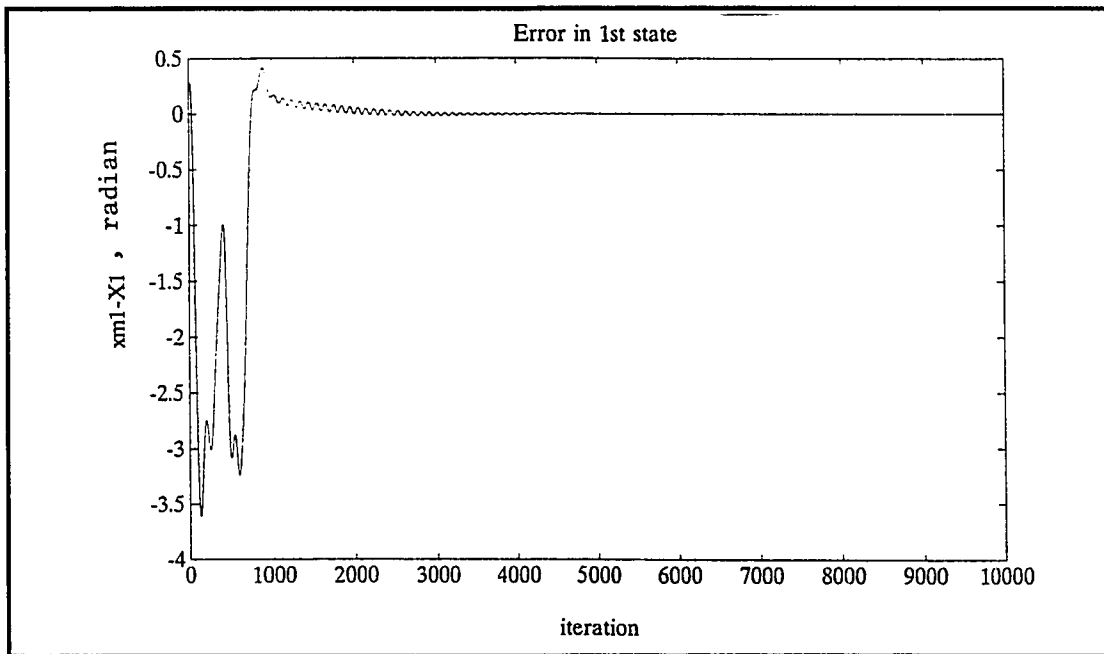


Figure 6.7b The error between the first linear and linearized state

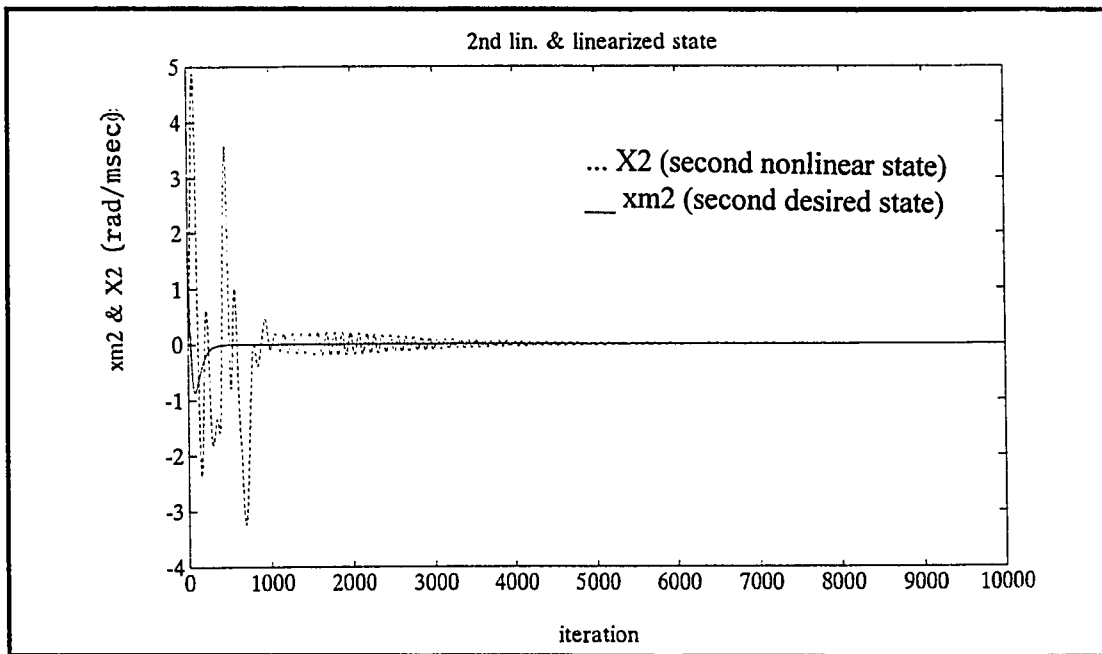


Figure 6.7c The second linear and linearized state with initial condition; $x_1=45^\circ, x_2=x_3=0$.

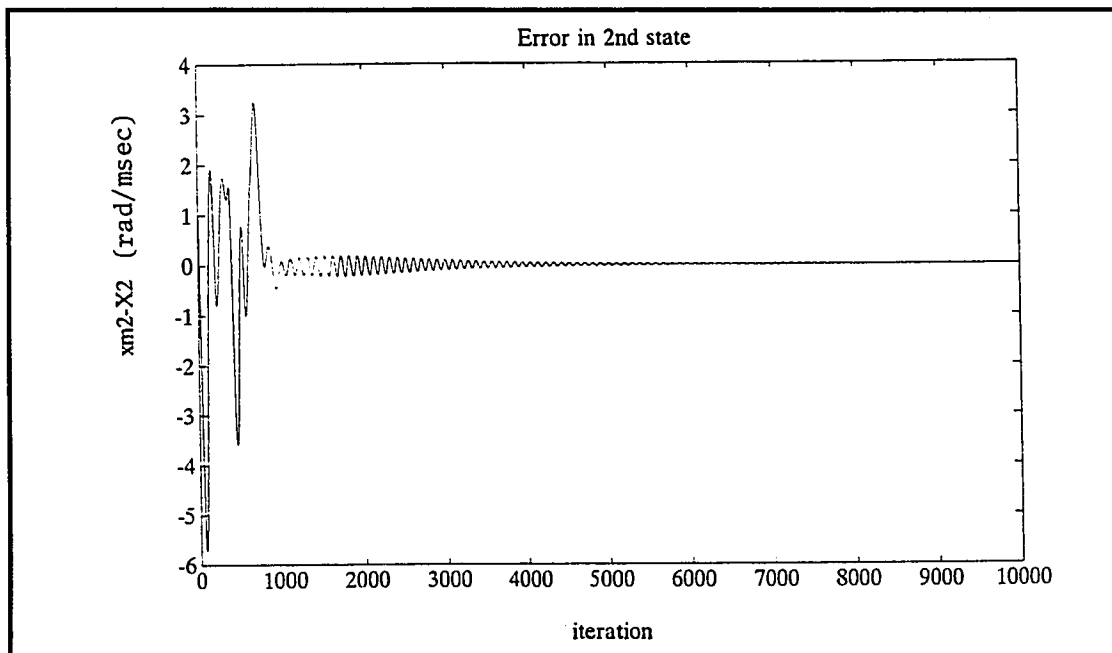


Figure 6.7d The error between the second linear and linearized state

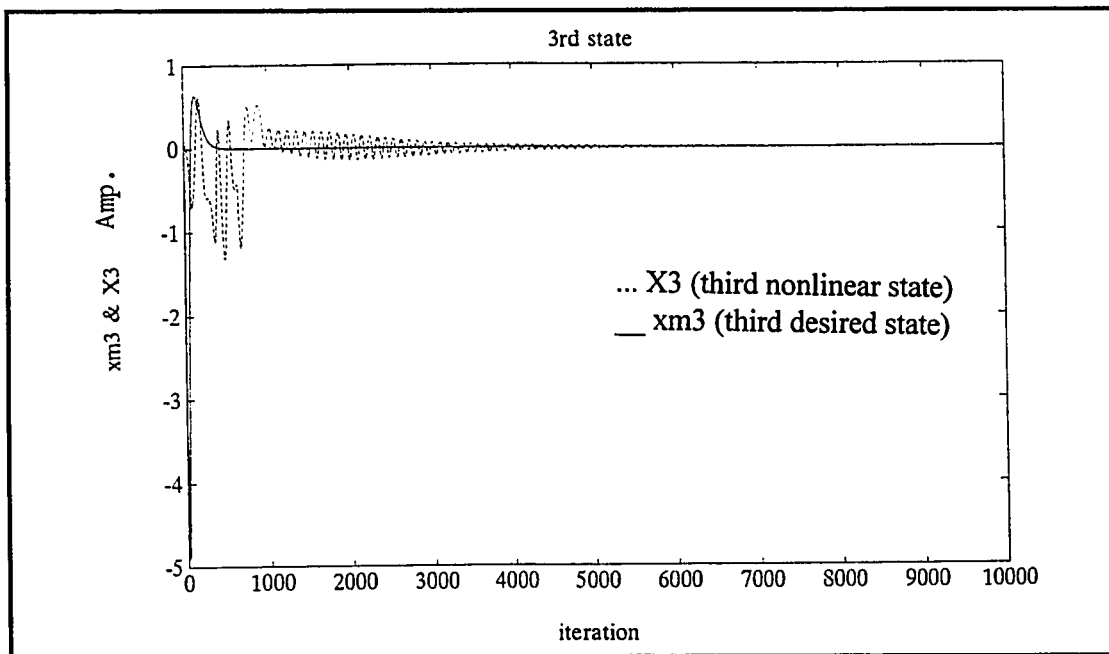


Figure 6.7e The third linear and linearized state with initial condition; $x_1=45, x_2=0, x_3=0$.

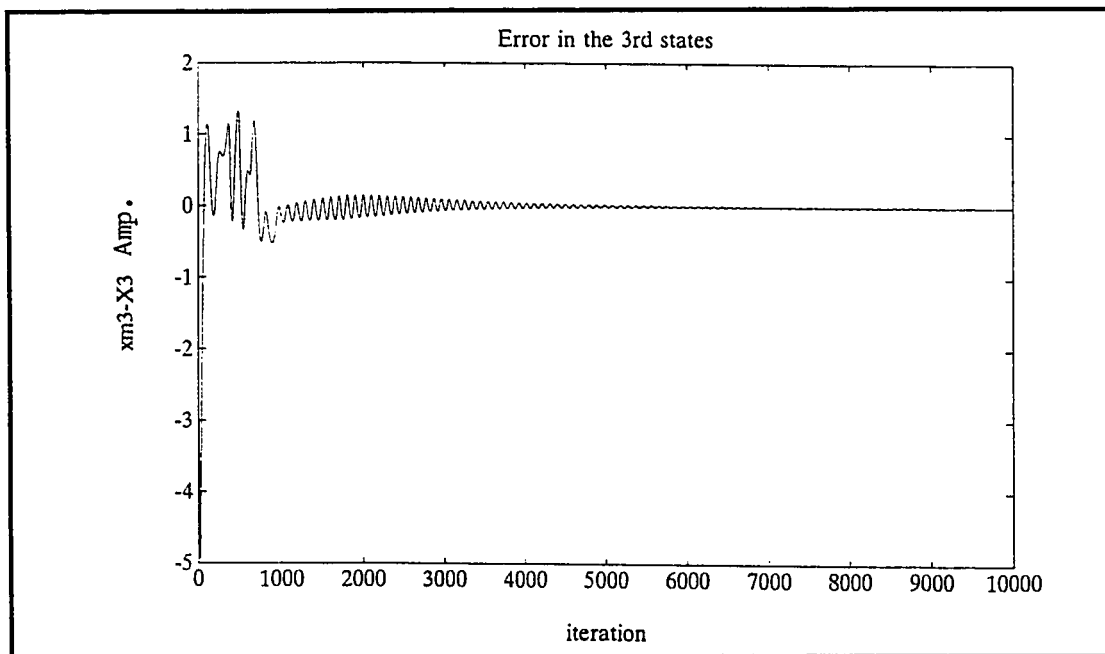


Figure 6.7f The error between the third linear and linearized state

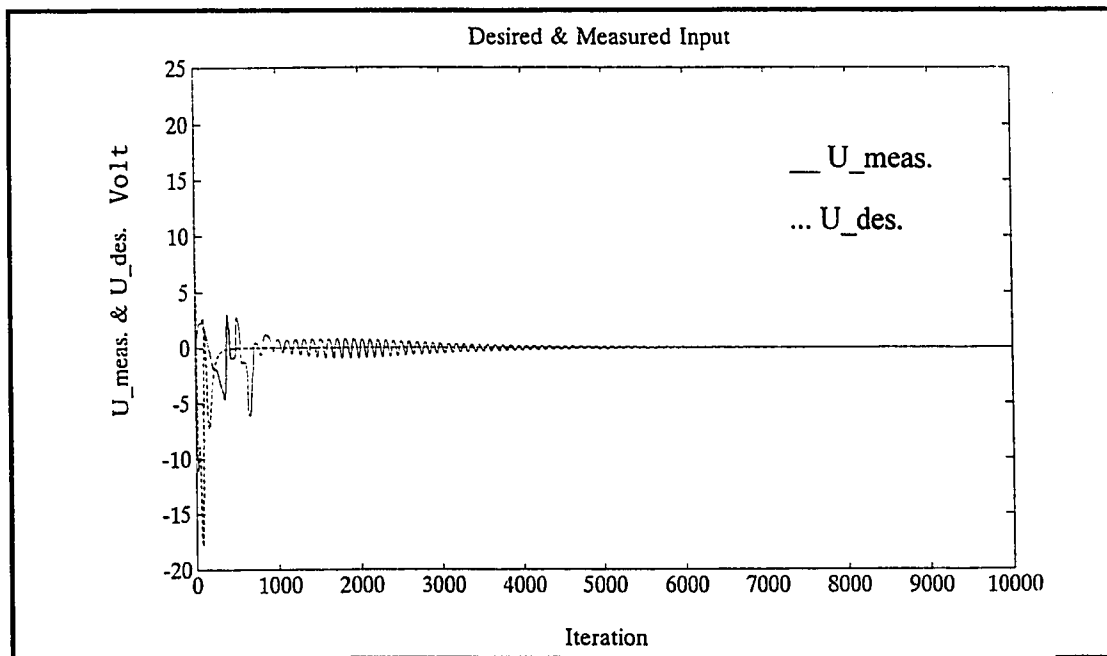


Figure 6.7g The desired linearizing input of the feedback linearization method and the linearizing input produced by the proposed NN controller required to stabilize the inverted pendulum.

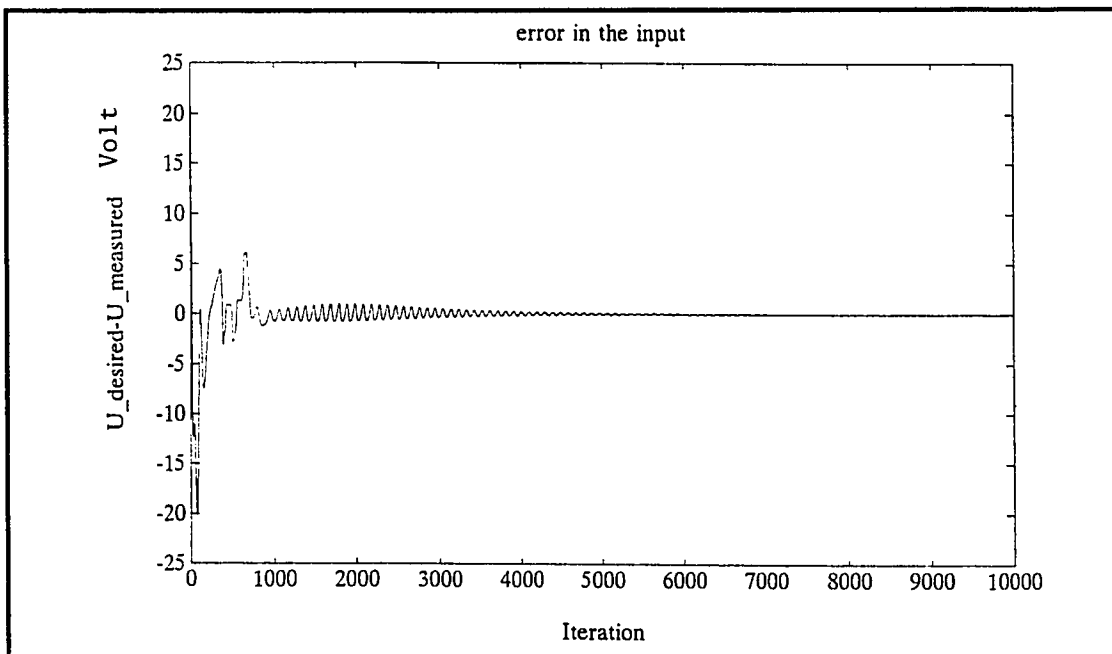


Figure 6.7h The error between the desired linearizing input calculated by feedback linearization and the measured linearizing input measured by NN.

6.3 The Van Der Pol Equation

The second order nonlinear differential equation:

$$m\ddot{x} + 2c(x^2 - 1)\dot{x} + kx = 0 \quad 6.34$$

with m , c , and k being positive constants, is the famous van der pol equation. It can be regarded as describing a mass-spring-damper system with a position dependent damping coefficient $2c(x^2 - 1)$, or equivalently, an RLC electric circuit with a nonlinear resistor.

In state space representation, the above dynamic can be described as:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -\frac{2c}{m}(x_1^2 - 1)x_2 - \frac{k}{m}x_1 \end{aligned} \quad 6.35$$

where $x_1 = x$, and $x_2 = \dot{x}$

This system has a unique unstable equilibrium point at the origin. Furthermore, it has a stable limit cycle. Limit cycle is defined as an isolated curve, and stable limit cycle means that all trajectories in the vicinity of that cycle converging to it as $t \rightarrow \infty$. Figure 6.8 shows for different values of c that the system has a unique closed curve that attracts all trajectories starting off the curve [48].

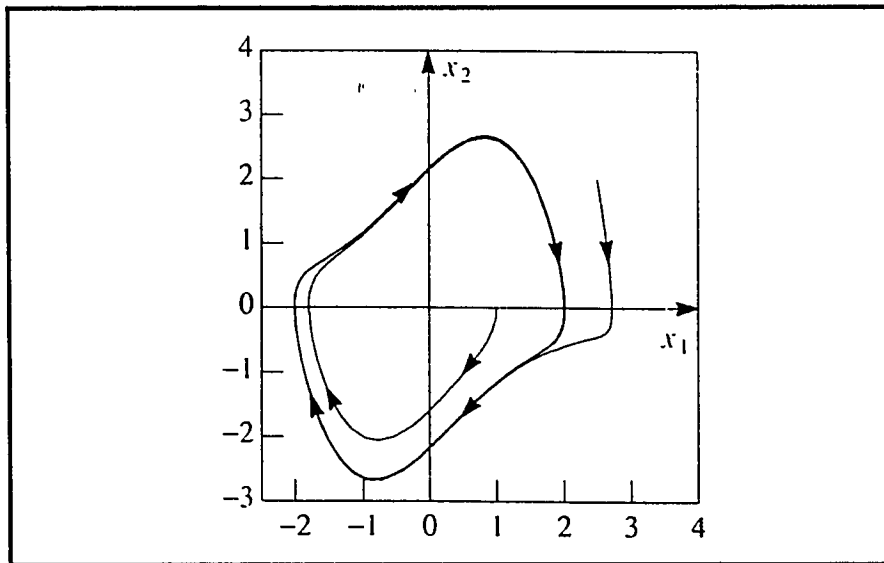


Fig. 6.8a

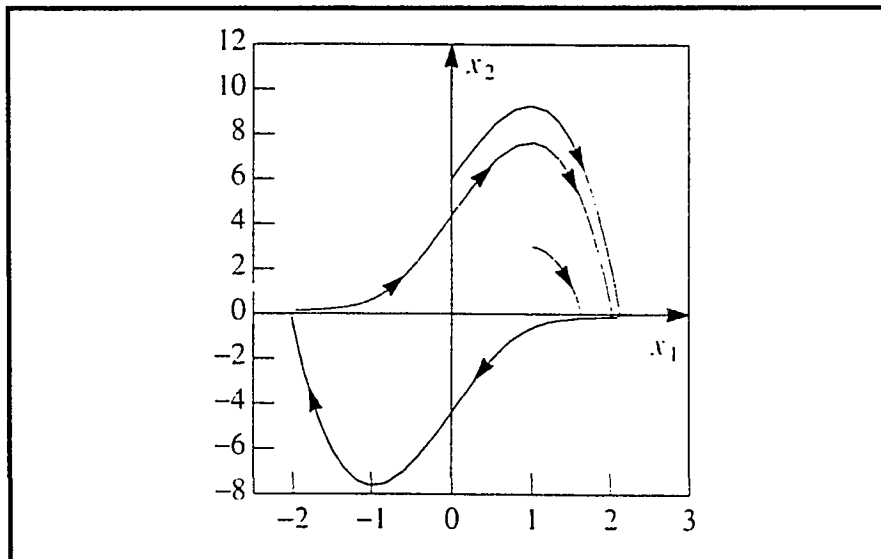


Fig. 6.8b

Figure 6.8 The limit cycle of the system.(a) $c=1$, (b) $c=2$

Our objective is to stabilize this system at the origin. In order to stabilize this system at this equilibrium point, we need to apply a control force. So, the above dynamic has the following form;

$$m\ddot{x} + 2c(x^2 - 1)\dot{x} + kx = u \quad 6.36$$

and the state space representation became;

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -\frac{2c}{m}(x_1^2 - 1)x_2 - \frac{k}{m}x_1 + \frac{1}{m}u \end{aligned} \quad 6.37$$

This system is clearly in the controllability canonical form. Introducing the linearizing input:

$$u = mv + 2c(x_1^2 - 1)x_2 + kx_1 \quad 6.38$$

leads to transform the nonlinear system into the following linear system:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= v \end{aligned} \quad 6.39$$

reasonable parameters of our system are; $c=1,2$, and $m = 1$, and $k = 1$.

Now, suppose that we would like the nonlinear system to follow the states of the linear time-invariant system with eigenvalues assigned at $\lambda_{1,2} = -2, -2$. Then, the characteristic equation of the closed-loop linearized system has the form $(s+2)(s+2)$.

The state feedback assigning the desired eigenvalues to the linearized closed-loop system is

$$v = -c_0x - c_1\dot{x} \quad 6.40$$

with the positive constants c_i 's chosen properly such that the polynomial $s^2 + c_1s + c_0$ has all its roots strictly in the left half plane. Thus, if the positive constants c_i 's chosen such that

$$(s+2)(s+2) \cong s^2 + c_1s + c_0 \quad 6.41$$

Then, the new input is

$$v = -4x_1 - 4x_2 + 4x_d \quad 6.42$$

and, the linear time-invariant system in (6.35) has the following form:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -4x_1 - 4x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 4 \end{bmatrix} r \quad 6.43$$

where r is the desired position, In this case $r = x_d = 0$.

With the neural network configuration given in table 3, we found that the proposed controller generates the desired input signal to the plant and the nonlinear states follow the desired states given by the system in (6.43).

#neurons in input layer	#neurons in 1st hid.layer	#neurons in 2nd hid. layer	#neurons in output layer
4	10	10	1

TABLE 3

Figure 6.9(a-f) illustrate the simulation results. Clearly, the proposed neural network produces the required input signal to drive the nonlinear states from its initial state to the desired ones after 9000 iterations.

The systems described by the Van Der Pol equations are oscillatory in nature and these oscillation are quite robust, meaning that starting from about any initial condition you always converge to the limit cycle. The feedback linearization method and feedback linearization through neural network will completely destroy these oscillations and produce a stable system that can be easily controlled by linear control method.

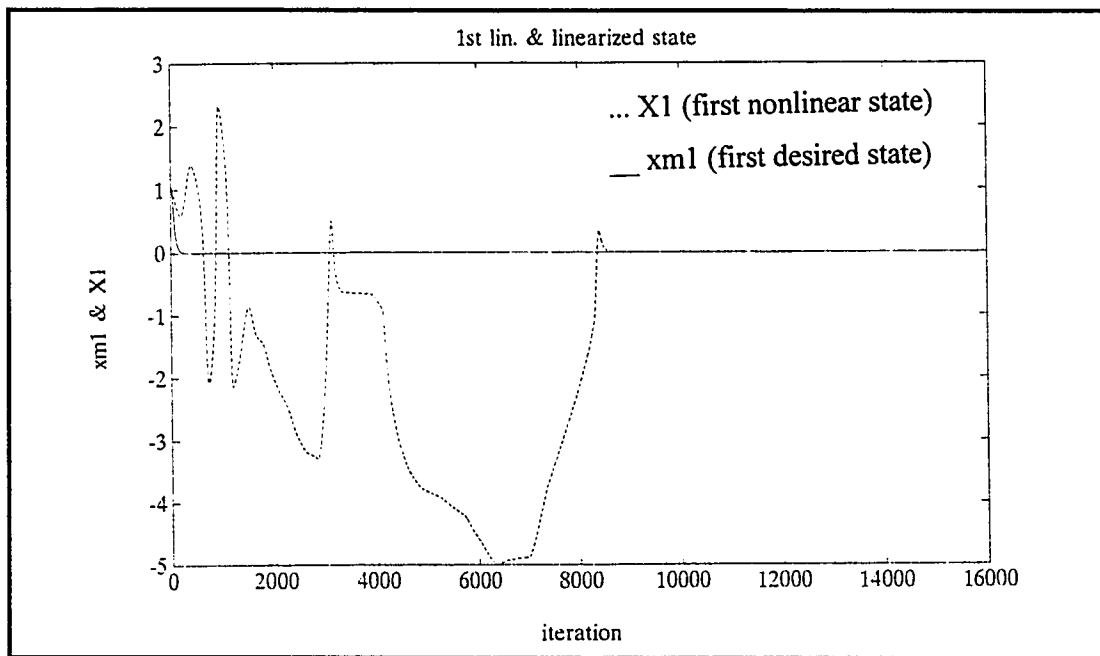


Figure 6.9a The first linear and linearized state with initial condition; $X=[1,0]$.

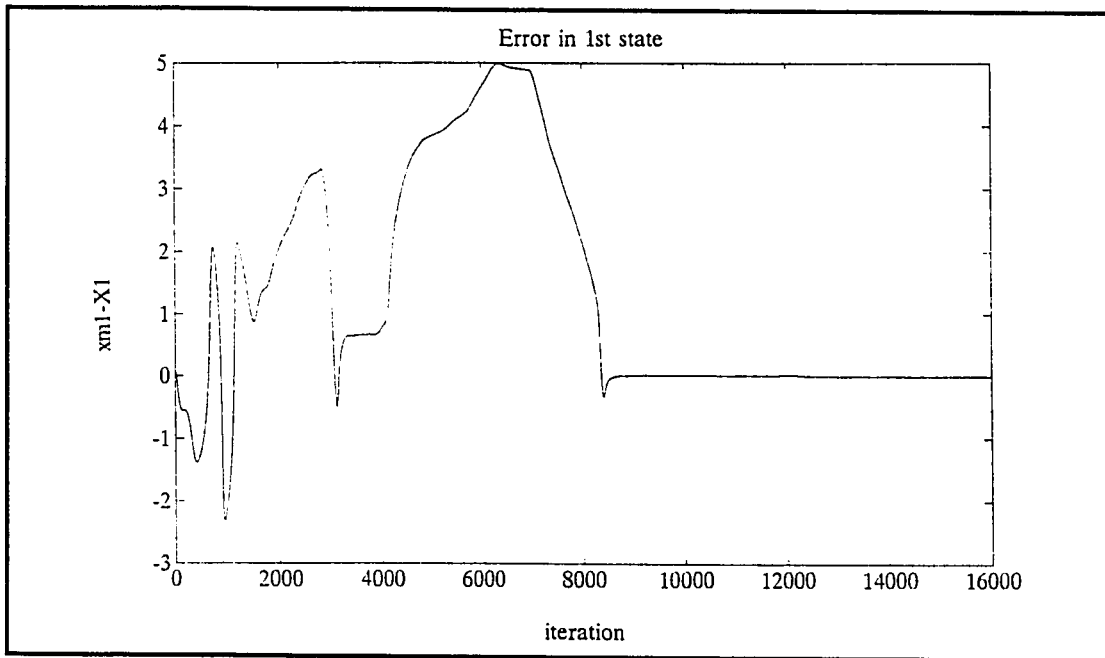


Figure 6.9b The error between the first linear and linearized state.

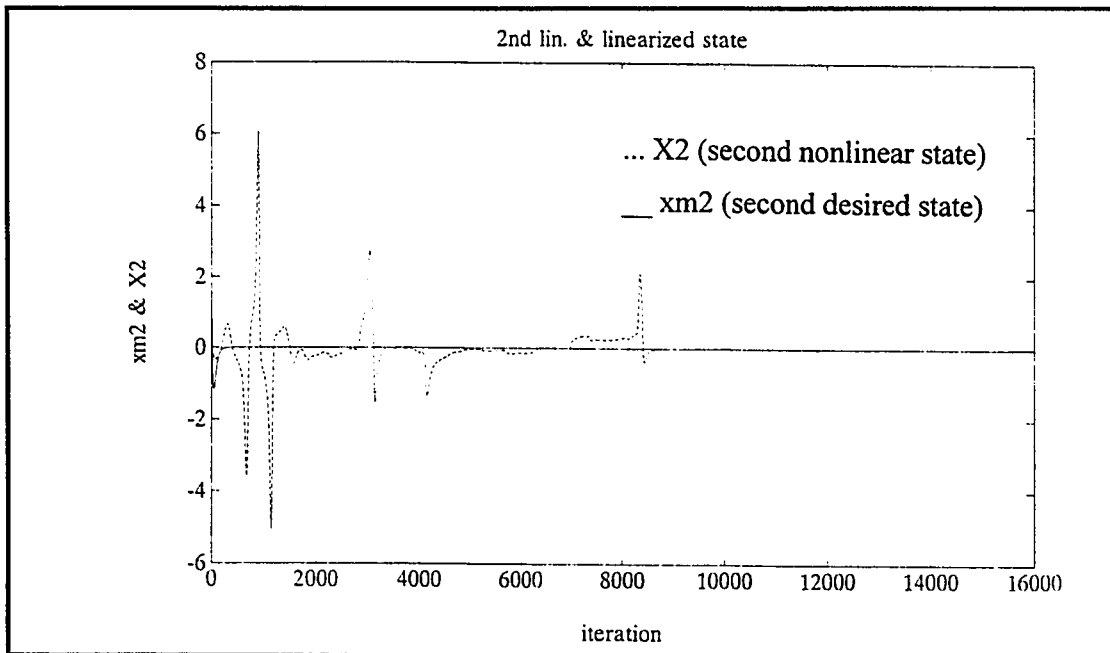


Figure 6.9c The second linear and linearized state with initial condition; $X=[1,0]$.

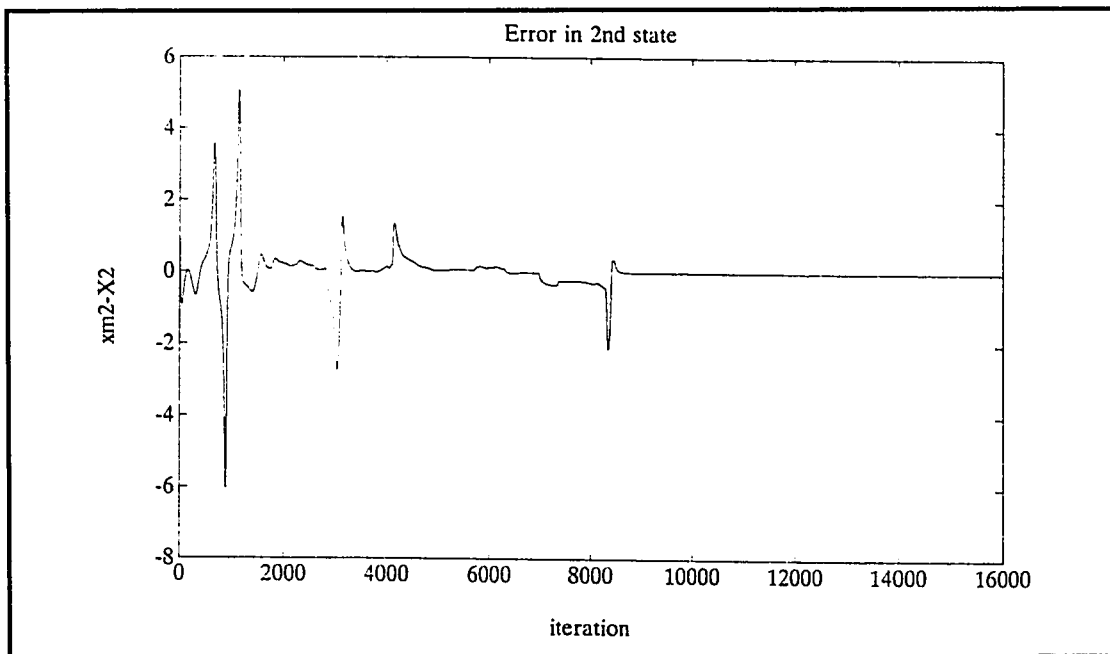


Figure 6.9d The error between the second linear and linearized state.

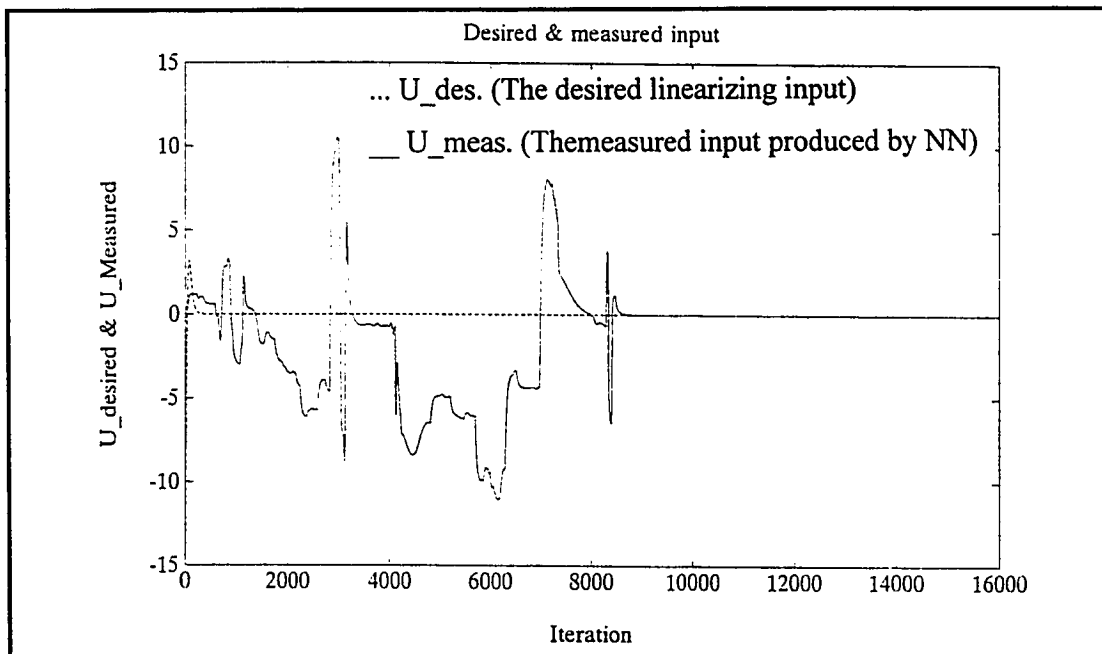


Figure 6.9e The linearizing input calculated by feedback linearization method and the linearizing input generated by the proposed NN.

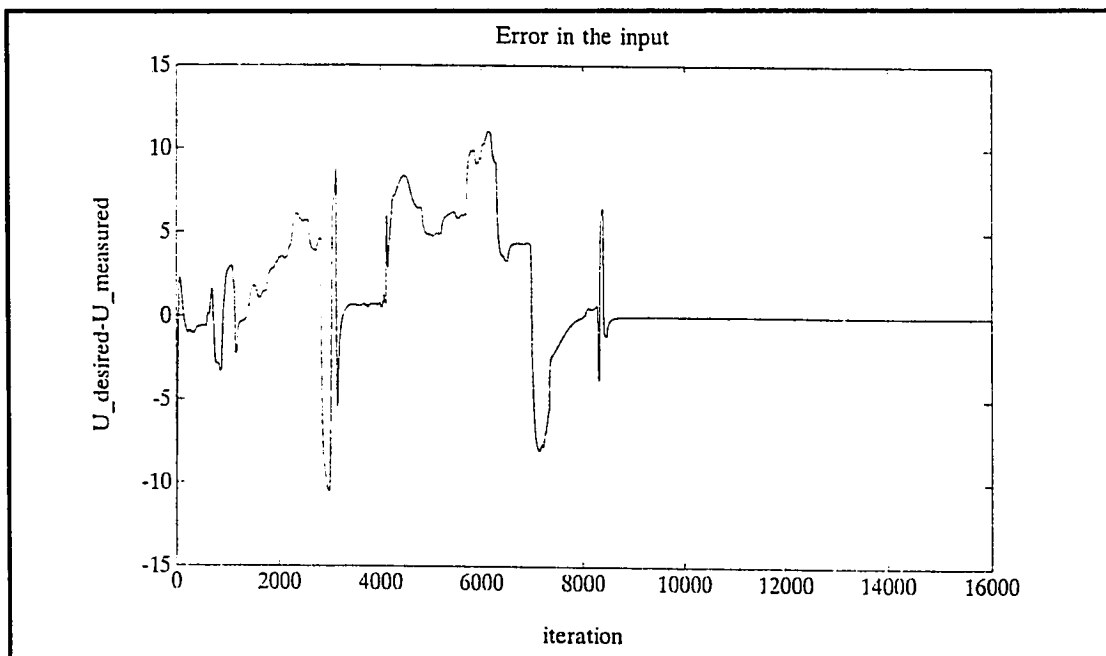


Figure 6.9f The error between the linearizing inputs of feedback linearization and the proposed NN.

CHAPTER 7

CONTRIBUTIONS,

CONCLUSIONS AND

RECOMMENDATIONS

An area of control systems which offer much scope for the use of artificial neural networks is nonlinear control. In this work we have concentrated upon a method to perform input-state feedback linearization using neural networks.

7.1 Contributions

The main contributions in this thesis can be summarized as follows:

1. The neural network architecture suggested by Levin and Narrendra in [2] is simplified. In the proposed scheme, one neural network proves its capability to perform input-state feedback linearization method.
2. The proposed scheme avoids the training stage and the identification step which is considered as the first essential step in [2].
3. The back propagation algorithm through the plant suggested by Psaltis et al. [10] is modified and implemented. Also, it is proved to be computationally efficient.

4. Input-state feedback linearization is performed on-line.
5. The state transformation is avoided.
6. Input-state feedback linearization is applied to unknown nonlinear systems.

7.2 Conclusions

1. Neural networks can be effectively used as a direct adaptive feedback linearization controller. That is to say, the neural networks are capable of generating the desired linearizing input required to drive the plant from its initial states to the desired states of a linear time invariant system.
2. The proposed neural network is simple in architecture and easier to implement.
3. Neural Network controller can perform input-state feedback linearization for unknown dynamics. Provided that they are feedback linearizable.
4. The proposed scheme is systematic. This is due to the fact that state transformation is avoided.
5. An overshoot occurs as a result of setting the initial weights to small random numbers.

7.3 Recommendations

1. Extension to systems which are not feedback linearizable can be considered. This can be done if the architecture of the proposed method is built according to dynamic feedback.
2. Extensions to MIMO case can also be considered.
3. Input-output feedback linearization with the existence of internal dynamics can be undertaken. (i.e., when the relative degree is less than the system's order).
4. Theoretical study can also be undertaken to analyze the proposed method in order to gain insight about local and global stability properties.

5. Comparisons can also be considered with other schemes using neural networks, such as neural network controller based on an indirect control methods, and neural networks based on learning using the forward and inverse dynamics method, or neural network with sliding control scheme.
6. Improving the transient response can be undertaken. This is can be accomplished if the initial weights are chosen in a different way rather than setting them randomly.
7. The slow convergence property can be improved by using different modified backpropagation algorithm, such as the backpropagation algorithm with adaptive learning rate and a momentum term.

Bibliography

- [1]A.J. Krener, "On the equivalence of control systems and the linearization of nonlinear systems," SIAM. J. Control, vol. 11, No. 4, pp. 670-676, November 1973.
- [2]A.U. Levin and K. S. Narendra, "Control of nonlinear dynamical systems using neural networks: Controllability and Stabilization," IEEE Tran. on Neural Networks, vol. 4, no. 2,pp. 192-206, March 1993.
- [3]B. Widrow and M.A. Lehr, "30 years of adaptive neural networks: Perceptron, Madaline, and Backpropagation," Proc. IEEE , vol 78, no. 9, pp. 27-51, Sept. 1990.
- [4]B. Jakubczyk and W. Respondek, "On linearization of control systems," Bull Acad Polonaise Sci. Ser. Sci. Math., vol. 28, pp 517-522, 1980
- [5]B. Widrow, Capt. R.G. Winter, and R.A. Baxter, "Learning phenomena in layered neural networks," Int. J. on Neuural Network, vol. 2, pp. 411-429, 1988.
- [6]C.-T. Chen, Linear system theory and design, Holt,Rinehart and Winston, 1984.
- [7]C. Batur, H. Zhang, J. Padovan, and V.S. Kasparian , "Davidon least squares based neural network learning algorithm," ACC 1992,pp. 973-977.
- [8]D.H. Nguyen and B. Widrow, "Neural networks for self-learning control system," IEEE Cont. Sys. Mag., pp. 18-23, 1990.
- [9]D.R. Hush and B.G. Horne, "Progress in supervised neural network, " IEEE Sig. Proc., pp. 8-37, Jan. 1993.
- [10]D. Psaltis, A. Sideris, and A. Yamamura, "Neural controllers," IEEE Cont. Syst. Mag., April 1988.
- [11]E. Mu and J.T. Gain, "Nonlinear systems linearization: A survey," Proceedings of the 21st Annual Pittsburgh Conference, pp. 2203-2208, May 1990.
- [12]F.C. Chen, "Backpropagation neural networks for nonlinear self-tuning adaptive control," IEEE Cont. Syst. Mag., pp. 44-47, 1990

- [13]F.C.Chen and C.C. Liu, "Adaptively controlling nonlinear continuous time systems using neural networks," ACC, pp. 46-50, 1992.
- [14]F.C.Chen and H.K. Khalil, "Adaptive control of nonlinear systems using neural networks," Proc. of 29th Conference on Decision and Cont., pp. 1707-1712, Dec. 1990.
- [15]F.C Chenand H.K. Khalil, "Adaptive control of nonlinear systems using neural networks-A dead zone Approach," ACC, pp. 667-672. 1991
- [16]G.A. Carpenter, "Neural network models for pattern recognition and associative memory," Neural Networks, vol. 2, pp. 243-257, 1989.
- [17]J. Chiasson and M. Bodson, "Nonlinear control of a shunt DC motor," IEEE Tran. on Aut. Cont., vol. 38, no. 11, pp.1662-1666,1993.
- [18]J.W. Grizzle and P.V. Kokotovic, "Feedback linearization of sampled data systems," IEEE Tran. on Aut. Cont., vol. 33, no 9, pp. 857-859, Sep. 1988.
- [19] J.-J. Slotine, Applied Nonlinear Control. Printice-Hall, 1991.
- [20]J. Zrida and A.A. Al-Ali, "A novel neural network adaptive controller structure for a resonant circuit," Proceedings of the 7th Mediterranean Electrotechnical Conference (MELECON'94), Antalya, Turkey, April 94.
- [21]K. Hornik, "Multilayer feedforward networks are universal approximators," Neural Networks, vol. 2, pp. 359-366,1989.
- [22]K..J. Hunt, D. Sbarbaro, R. Zbikowski, and P.J. Gawthrop, "Neural networks for control systems-A Survey," Automatica, vol. 28, no. 6,pp. 1083-1112, 1992.
- [23]K.S. Narendra and K. Parthasarathy, "Gradient methods for the optimization of dynamical systems containing neural networks," IEEE Tran. on Neural Net., vol. 2, no. 2, pp. 252-262, 1991.
- [24]K.S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," IEEE Tran. on Neural Net., vol. 1, no. 1, pp.4-27, 1990.

- [25]L.G. Kraft and D.P. Campagna, "A comparison between CMAC neural network control and two traditional adaptive control systems," IEEE Cont. Syst. Mag., pp. 36-43, 1990.
- [26]L. Jin, P.N.Nikiforuk, and M.M. Gupta, "Adaptive tracking of SISO nonlinear systems using neural networks," ACC, pp.56-60, 1992.
- [27]L.R. Hunt, R. Su, and G. Meyer, "Global transformations of nonlinear systems," IEEE Tran. on Aut. Cont., vol. 28, no. 1, pp. 24-30, Jan. 1983.
- [28]M.A. Waddoups and K.L. Moore, "Neural networks for iterative learning control," ACC, pp. 3049-3051, 1992.
- [29]M. Fukumi and S. Omatu, "A new backpropagation algorithm with coupled neuron," IEEE Tran. on Neural Net., vol. 2, no. 5, pp. 535-538, Sep. 1991.
- [30]M. Fliess, "Generalized controller canonical forms for linear and nonlinear dynamics," IEEE Tran. on Aut. Cont., vol. 35, no. 9, pp. 994-1000, Sep. 1990.
- [31]M. Saerens and A. Soquet, "Neural controller based on backpropagation algorithm," IEEE Proc., vol. 138, no. 1, pp. 55-62, Feb. 1991.
- [32]M.S.Lan, "Adaptive control of unknown dynamical systems via neural network approach," ACC, pp. 910-915, 1989.
- [33]P.D. Olivier, "Feedback linearization of DC motors," IEEE Tran. on Ind. Electronics, vol. 38, no. 6 pp. 498-501, Dec. 1991.
- [34]R. Sommer, "Control design for multivariable nonlinear time varying systems," Int. J. Cont., vol. 31, no. 5, pp. 883-891, 1980.
- [35]R. Su, "On the linear equivalents of nonlinear systems," Syst. and Cont. letters, pp. 48-52,1982.
- [36]R.P. Lippmann, "An introduction to computing with neural nets," IEEE ASSP Mag., pp. 4-22, 1987.

- [37]E. Freund,"The structure of decoupled nonlinear systems," *Int. J. Cont.*, vol. 21, no. 3, pp. 443-450,1975.
- [38]S.A. Al-Baiyat and A.H. A. Rahim, "Dynamic bracking resistor-reactor switching strategies through a novel linear transformation technique," *Elec. Mach. and Power Syst.*, 21, pp. 543-555, 1993.
- [39]S.H. Zak and C.A. Maccarley, "State feedback control of nonlinear systems," *Int. J. Cont.*, vol. 43, no. 5, pp. 1497-1514, 1986.
- [40]W. Lie and J.J.E. Slotine, "Neural network control of unknown nonlinear systems," *ACC*, pp. 1136-1141, 1989.
- [41]W.A. Porter," Decoupling of and inverses for time time-varying linear system," *IEEE Tran. on Aut. Cont.*, pp. 378-380, Aug. 1969.
- [42]X. Demin, J. Dehun, and R. Zhang, "An improved neural controller architecture," to be published.
- [43]R.W. Brockett,"Feedback invariants for nonlinear systems," in *Proc. 6th IFAC Congress, Helsinki, 1978*, pp. 1115-1120.
- [44]T. J. Tarn, A.K. Bejczy, A. ISidori and Y. Chen," Nonlinear Feedback in robot arm control," in *Proc. 23rd IEEE conference on Decision and Cont.*, Las Vegas, NV, Dec. 1984.
- [45]G. Meyer, R. Su and L. R. Hunt," Application of nonlinear transformations to automatic flight control," *Automatica*, vol. 20, no. 1, pp. 103-107,1984.
- [46]Y. Arkun and J. calvet," Feedforward and feedback linearization of nonlinear systems and its implementation using internal model control (imc)," *Industrial and Chemistry Research*, vol. 27, no. 10, pp. 1822-1831, 1988.
- [47]M. Hagan, *Neural Network toolbox*, pp. 8.1-8.11.
- [48]H.K. Khalil, *Nonlinear Systems*, Macmillan publishing company,1992.

[49]A. Isidori, *Nonlinear Control Systems: An Introduction*. New York: Springer-Verlag, 1985.

Vita

- K. A. S. Al-Sahli
- Born in Al-Madina Al-Monawarah, 1963
- Received Bachelor's degree in Electrical Engineering from King Fahd University of Petroleum and Minerals in 1987.
- Worked as an officer at the Air Defence, Saudi Arabia.
- Joined Electrical Engineering Department at KFUPM in 1992.
- Completed Master's degree requirements at KFUPM, Dhahran, Saudi Arabia in 1995.