Investigation of Feedforward Neural Networks and Its Applications to Some Nonlinear Control Problems

NG Chi-fai

A Thesis Submitted in Partial Fulfillment

of the Requirements for the Degree of

Master of Philosophy

in

Mechanical and Automation Engineering

©The Chinese University of Hong Kong

December 2000

Abstract

Neural networks represent an emerging technology which has been widely used in many engineering disciplines. They are endowed with some unique attributes: universal approximation, the ability to learn from and adapt to their environment, and the ability to invoke weak assumptions about the underlying physical phenomena responsible for the generation of the input data. An important class of neural networks: multilayer perceptrons, also called feedforward neural networks, perform efficiently towards the design of the widest range of successful forecasters, pattern classifiers, controllers and sensors. Feedforward neural networks can act as a *universal approximator*. It can not only be used to approximate nonlinear mappings, but also be used to solve many complex nonlinear equations including nonlinear partial differential equations, up to any degree of accuracy. This capability has been explored in control community to solve various difficult nonlinear control problems. This thesis will explore the capabilities of the feedforward neural networks in two different ways. On the one hand, we will study the feasibility of using the neural networks to approximate a nonlinear control law arising from the feedback linearization approach. The motivation of doing so is that the control law based on the feedback stabilization is usually highly nonlinear, and is difficult to be implemented in real time. The mass processing capability of a feedforward neural network due to its parallel structure lends itself to an efficient way of computation. Our investigation has been applied to a flexible joint robot control system. A nonlinear control law is first synthesized based on the feedback linearization approach. The resulting control law is then approximated by a feedforward neural network through a supervised learning approach. Simulation study shows that the system performance driven from the neural based control law is very close to the real system performance which strongly supports the argument of the feedforward neural network approximation ability.

On the other hand, we have pursued a more difficult problem of using the neural networks to approximately solve a set of nonlinear partial differential equations known as center manifold equations. The center manifold equations arise from many nonlinear control problems such as the nonlinear output regulation problem. Due to the nonlinear nature, it is difficult to obtain the closed form solution of the center manifold equations. We have developed an approximation approach to solving the center manifold equations based on the universal approximation theorem of the feedforward neural networks. Many examples have been studied to test the effectiveness of our approach. In particular, we have applied our approach to two well known nonlinear control problems, namely, the asymptotic tracking problem of the ball and beam system and the disturbance rejection problem of the translational-oscillational-rotational-actuator (TORA) system. Our designs have been compared with the existing approach which is based on the Taylor Theorem. Computer simulation shows that our approach is superior to the Taylor series based approach in most of the cases. Additionally, we also studied the effects of other factors to the performance of our design. These include the effects of different activation functions to the approximation accuracy, and convergence of the weights, and the effects of the different feedback gains on the steady state tracking errors.

In summary, our investigation show that neural networks can indeed be used to solve many nonlinear control problems with a better solution than conventional approaches.

摘 要

神 經網 絡這一新 興技 術已被廣 泛應 用於許 多工 業領 域。它具 有以 下特性：通用逼 近、自 學及適 應環 境的能 力。其 中，多層 PERCEPTRON，也稱前饋神 經網 絡，能成 功實 現多 方 面預 測、模 式分 類、 控 制器及傳 感 器的設計。前饋神 經網 絡可用作通用逼 近，它不 僅可 以逼 近非線性映射，也可用來解 決許 多複 雜的非線 性方 程包 括非線 性偏微 分方 程。前饋神 經網 絡的這一特 性已被用于控 制領 域中解 決許多非線性控 制問 題。本文擬從兩個不 同角 度探 討了前饋神 經網 絡的能 力。

一方 面，基 于反 饋線 性化的控 制律通 常是非線 性的，因而難 以實 現實時控 制，我們將研 究使 用神 經網 絡來逼 近由反 饋線 性化方 法產 生的非線 性控 制器。由 于前饋神 經網 絡具 有並 行結 構，因 此它有很 強的處 理能 力，是一有 效的計 算方 法。我 們的方 法已被用 于具 有柔性關 節的機 器 人控 制系 統中。首先，由反 饋線 性化方 法綜 合設 計出一非線 性控 制律；然 後用前饋神 經網 絡逼近該控 制律。仿 真結 果表 明由基 于神 經網 絡的控 制律控 制的系 統，其性 能非常接 近實 際系 統的性 能。

另一方 面，我 們將研 究更為複 雜的問 題：用神 經網 絡近 似解一組非線 性偏微 分方 程(中 心 流形方 程)。包 括非線 性輸 出調節問題在 內的許 多非線 性控制問題都可歸 結為求解中 心 流形方 程。由 于其非線 性特 性，中 心 流形方 程的精 確解往 往難以解出。我 們研 究了一 種基 于通 用逼近理 論的近 似解法，並應用到許 多實例中包 括著 名的 BALL & BEAM 系 統的跟 蹤問 題和 TORA 系 統的抗干 擾問 題。計 算仿 真結 果表 明我 們的方 法优于現 有的其 他方 法。

總 之，我 們的研 究表 明神 經網 絡在解 決許 多非線 性控 制問 題方 面优與傳統的方 法。

Acknowledgments

I would like to take a few words on expressing my appreciation, with my deep heart, for those involved in the completion of this thesis. Within these two years, may I first thank my supervisor, Prof. J. Huang, for his kind guidance and supervision especially his financial support for my first year of postgraduate study. I have gained a lot of knowledges that are not only applicable to my research but also useful for my career. I would also like to thank Prof. Y.S. Xu and Prof. J. Wang, who have offered me valuable suggestions and advices as members of my thesis committee.

Apart from them, my colleagues have played an important roles on my progress. I often recall the scenes of our discussion on the problem of our researches. Sometimes, we are excited about certain arguments due to different point of view but the outcomes are always positive. I will never forget the faces and the smile of Jin Wang, Dan Wang, Jijun Zhao, Kin Yeung, WaiWing Law, ZhiYong Chen, Stephen Yeung, Thomas Lei, Mark Wong, WeiKeung Fung, Ricky Lam, Terry Ho, River Wong, Clifford Chow and Winston Sun.

At last, I would like to thank my family who have paid a lot of support on me.

Again, thanks for all of them. This thesis can not be completed without their support and friendship.

ChiFai Ng

Autumn 2000

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Objectives

The artificial neural network has been a powerful computational tool for solving many complex engineering problems. There are many books published for describing all kinds of neural networks and its applications [18],[15],[46],[44]. The mathematical representation of a neuron began with the pioneer scientist McCulloch and Pitts at 1940s. In late 1950s, a single layer perceptron was invented by Rosenblatt's. After 30 years of continuation of research in this field, the multilayer feedforward neural network was developed in conjunction with the back-propagation algorithm by Rumelhart, Hinton and Williams in 1986. It was the first time that the engineering communities was provided with an effective methodology for the construction of real nonlinear systems accepting large number of inputs and resulting in remarkable success in applications to engineering problems of classification, regression and forecasting. The development of back-propagation algorithm represents a milestone in neural networks in that it provides a computationally efficient method for the training of multilayer perceptrons.

1.2 Principles of Feedforward Neural Network Approximation

The feedforward neural networks consists of a group of neurons, the basic unit of the neural network, that are arranged layer by layers [18]. A typical feedforward neural network, shown

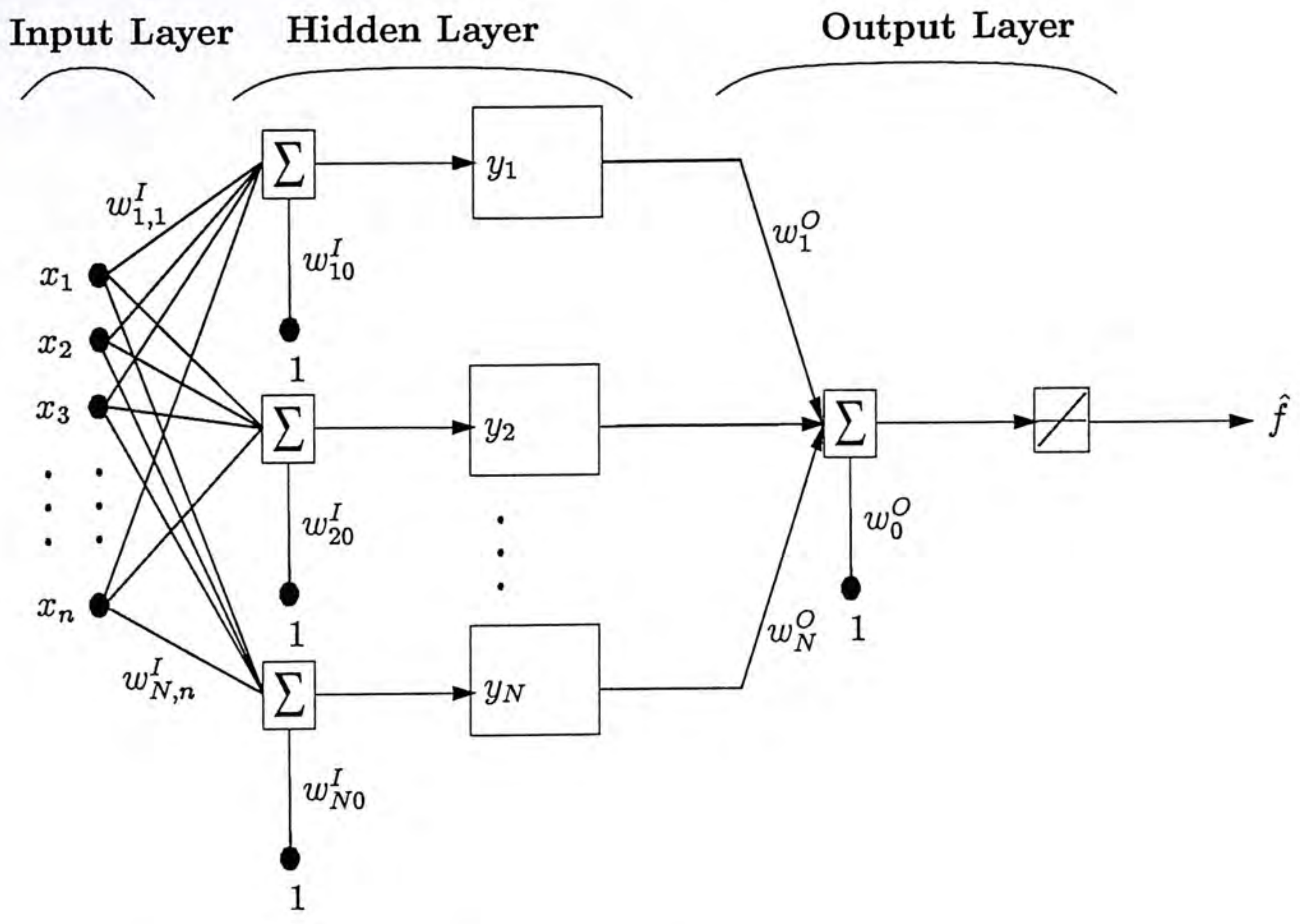


Figure 1.1: A model of three-layer Feedforward Neural Network

in Figure 1.1, has an architecture of three different layers: input layer, hidden layer and output layer. The network is fully connected in the direction from input layer to hidden layer to output layer which means that a neuron in any layer of the network is connected to all the nodes/neurons in the previous layer. There is a parameter, called weight, which is connected between two nodes from adjacent layer. Each neuron, stored a function or called activation function, gives its output value with respect to the characteristics of the function itself and the weighted inputs. Consequently, the output signal is generated by the transmission of input signals associated with the activation functions stored in the neurons and the weights. In other words, it can be modeled as a function mapping with respect to the inputs, the weight value and the activation functions as follows:

$$\hat{f} = \Sigma_{i=1}^{N} w_i^O \phi(\sum_{j=1}^{n} w_{ij}^I x_j + w_{i0}^I) \tag{1.1}$$

Researches on the properties and capabilities of feedforward neural network had been well developed at late 1980s [20],[11]. The paper written by Hornik showed the approximation capabilities of a feedforward neural network upon any continuous functions. That successful result was announced in the paper which can be rephrased as follows [50]: given a real valued function $f \in C^k$ defined on a compact subset $\Gamma \in R^n$, and any $\epsilon > 0$, there exists an integer

N, and real numbers w_i^O, w_0^O, w_{ij}^I and w_{i0}^I, $i = 1, \cdots, N$, and $j = 1, \cdots, n$ such that the mapping defined by

$$\hat{f}(W, x) = \Sigma_{i=1}^N w_i^O \phi(\sum_{j=1}^n w_{ij}^I x_j + w_{i0}^I) \tag{1.2}$$

satisfies

$$\sup_{x \in \Gamma} |\hat{f}(W, x) - f(x)| < \epsilon \tag{1.3}$$

In (1.2), x_i, $i = 1, \cdots, n$, is the input, f the output, and the integer N the number of hidden layers. The real numbers w_i^O, w_{ij}^I, and w_{i0}^I, $i = 1, \cdots, N$, and $j = 1, \cdots, n$ are called weights. Specifically, w_i^O, $i = 1, \cdots, N$, is called the weight of the hidden neuron i, and w_{ij}^I, and w_{i0}^I are the weights and the bias term of the input neuron i, respectively. ϕ is sigmoidal activation function. The mapping defined by (1.2) is called three layer feedforward neural network. Figure 1.2 shows the common activation functions used in the feedforward neural

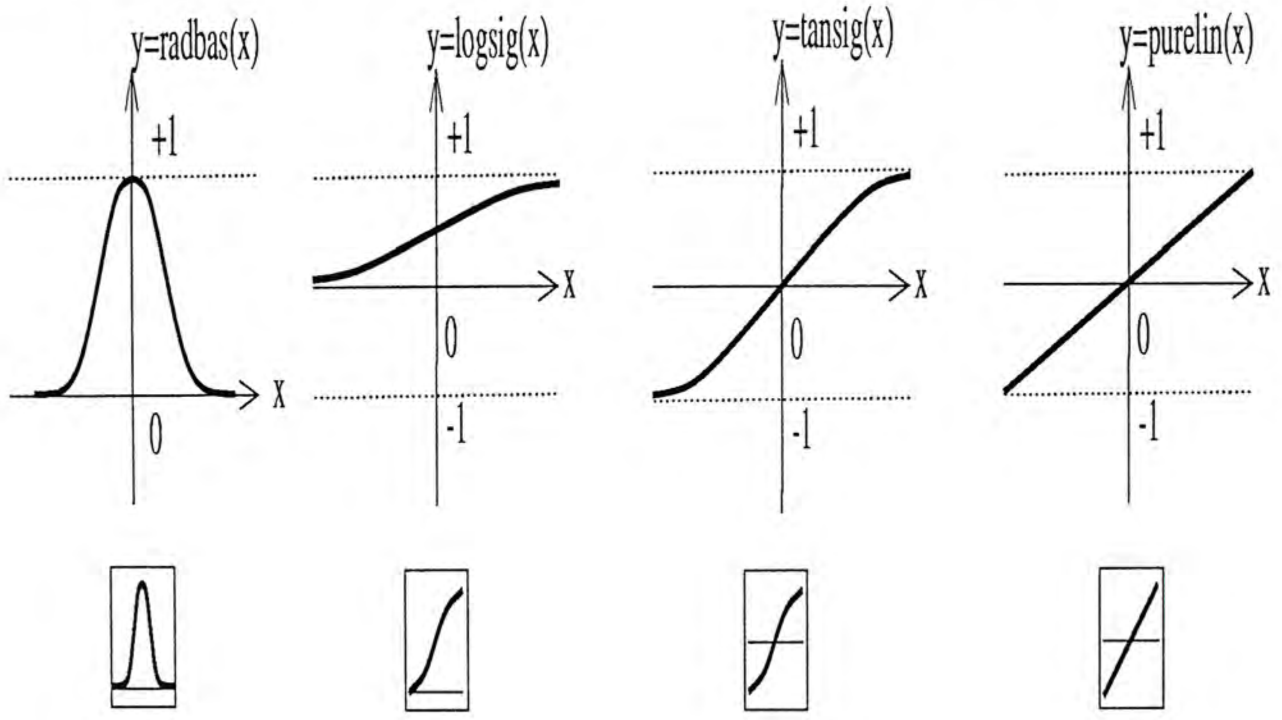


Figure 1.2: Common Activation Functions

networks. Usually, we choose the first three functions as the activation function in the hidden layer, and the linear function is used in the output neurons such that the overall mapping is nonlinear which can be performed in all real values. This network can be used as a general function approximator. It can approximate any function with a finite number of discontinuities, arbitrarily well, given sufficient neurons in the hidden layer. The outputs of the network have been substituted into an objective function which is minimized by updating the weights iteratively till the performance value of the objective function is less than ϵ.

A typical three layers neural network model is shown in Figure 1.1. The solvability of a nonlinear function is converted into a parameter optimization problem, that can be handled by gradient based methods. This method can not only lead to a nonlocal approximation scheme, but also offer some well known advantages associated with neural networks such as computational efficiency, and hardware realizability.

Research based on the approximation capability of the neural networks has been explored in many different engineering disciplines since the last decade. Many research results after 1990 have been done for utilizing the feedforward neural network approximation in solving the nonlinear control problems[50],[41],[2],[51],[10],[42],[38]. Some researches were concentrated on solving the nonlinear equations and functions such as [21],[19],[11]. Some other researches focus on improving the efficiency of the feedforward neural network approximation algorithm. [9],[7],[8],[37].

In this thesis, our interest is to handle some nonlinear control problems incorporating the feedforward neural network approximation techniques. Since 1980s, many nonlinear control approaches have been developed including feedback linearization, input-output linearization, adaptive control, sliding control, nonlinear H_∞ control, output regulation, etc [39], [31], [16], [35], [6]. A common feature of these nonlinear control approaches is that the synthesis of the control law involves complex computation. In many cases, it is either inefficient or in feasible to apply these methods to complex engineering systems. For example, the control law based on the feedback linerization is typically highly nonlinear, and is difficult to be implemented in real time. Thus it is interesting to consider using the neural networks to approximate this nonlinear control law so that the real time implementation of the control law is possible. As another example, it is known that the solvability of the output regulation problem relies on a set of partial differential and algebraic equations known as regulator equations[32],[27],[28],[29],[12],[22]. For a large class of nonlinear systems, the regulator equations can be reduced to the so-called center manifold equation, which is a set of nonlinear partial differential equations. Previous approach to solving the center manifold equations is based on the classical Taylor series approximation. This approach is less efficient, and is valid only in a neighborhood of the origin of the pertinent Euclidean space.

In this thesis, we will also explore using the feedforward neural network to solve the center manifold equations. .

1.3 Contribution of The Thesis

The research outcome of this thesis can be summarized as follows

- The study of the capabilities of the feedforward neural networks to approximate nonlinear mappings with application to the approximation of a feedback linearization based control law for a flexible joint robot system.
- The study of the neural network to solve the center manifold equations.
- Based on the approximated solutions of the center manifold equation, we design the nonlinear control laws for the ball and beam system and TORA system.
- The study of the effect of various feedback gains on the steady state response of the control law based on the output regulation theory.

1.4 Outline of The Thesis

The rest of this thesis is organized as follows:

- In chapter 2, we first study the multilayer feedforward neural networks. The brief description of the neural network and its learning algorithm are included. There are two main learning algorithms for training the neural networks: supervised learning and unsupervised learning. The free parameters, also called weights, are adapted by the steepest gradient descent method. We also brief some pros and cons for feedforward neural network approximation. For the supervised learning, we will implement a feedback linearization based control law of a flexible-joint robotic system. In the example, the nonlinear control law is designed by feedback linearization method. Assuming all the target states and initial states of the system are known, we will apply the supervised learning approximation method. Both the robot system and the controller have been approximated by respective three layer feedforward neural network.

Finally, the closed-loop robot system is approximated by combining two individual neural networks.

- Chapter 3 studies the approximation of center manifold equations. Due to the nonlinear nature, the closed form solution is not easy to obtain. An approximation method for solving the equation is thus proposed. Since it is a function approximation problem without knowing the target values of this equation, we need to apply the unsupervised approximation to solving this equation according to the universal approximation theorem. The objective function is defined in terms of the center manifold equations which is minimized by changing the values of a set of scalars, called weights of the feedforward neural network, iteratively. In other words, the problem of solving the center manifold has been converted into parameter optimization problem. An example has been used to demonstrate the effectiveness of our approach.

- In chapter 4, we will introduce the nonlinear output regulation problem, and connect its solvability to the solution of the center manifold equations. The objective of the output regulation problem is to design a control law which results in the asymptotic tracking of a reference input or/and the rejection of a disturbance. Both the reference input and the external disturbance come from the exosystem. The solvability of the problem boils down to the solvability of the regulator equations which can be further reduced to the solution of the center manifold equations for a large class of nonlinear systems.

- In chapter 5, an application to the asymptotic tracking of the ball and beam system is presented. The design of the control law is such that the output of the system is able to track the sinusoidal input asymptotically. The mathematical model of the system is described in which the center manifold equations can be derived. The objective function is defined with respect to the center manifold equations which will be minimized by our proposed approach with an arbitrarily small error. Two classes of feedback gain design methods are investigated and compared, namely, ITAE and Bessel prototype designs. It is shown, in comparison with the 1^{st} order and 3^{rd} order Taylor series approximated approach, that our approach led to a much smaller steady state error.

- In chapter 6, we will further apply our neural based approach to solve the disturbance rejection problem for TORA (Translational-Oscillational-Rotational-Actuator) system. This is a difficult control problem known as the nonlinear benchmark problem [23], [48], [49]. The design objectives are to have the output of the system able to reject the sinusoidal disturbance and keep the position of the cart asymptotically stable. We need to derive those pertinent center manifold equations, and solve it with the neural network approximation method. Similar to the previous example on the ball and beam system, a three layer feedforward neural network is adopted and the objective function is constructed based on the center manifold equations. The overall control law is obtained based on the approximation solution and a stabilizing feedback gain. Again, simulation results have been compared with the 1^{st} order and 3^{rd} order Taylor based approach. It is seen that the output of the system from the neural based approach attain a far smaller error than the linearized case. On the other hand, our neural based result is more or less the same as the 3^{rd} order one.

- In chapter 7, we draw some conclusions from our research and give some future research directions.

Chapter 2

Feedforward Neural Networks: An Approximator for Nonlinear Control Law

2.1 Optimization Methods Applied in Feedforward Neural Network Approximation

To utilize the approximation capability of a neural network, it is necessary to allow the network itself to learn in order to get closer to the desired function by adjusting the weight value iteratively according to the learning algorithm. Learning algorithms used in feedforward neural network approximation mainly fall into two broad categories: *supervised learning and unsupervised learning.*

In supervised learning, the learning rule is provided with a set of examples (the training set) of proper network behavior:

$$p_1, t_1, p_2, t_2, \ldots, p_n, t_n$$

where $p_i, i = 1, \ldots, n$, is an input to the network, and $t_i, i = 1, \ldots, n$, is the corresponding correct (target) output. As the inputs are applied to the network, the network outputs are compared to the targets. The learning rule is then used to adjust the weights and biases of the network in order to move the network outputs closer to the targets.

In unsupervised learning, the weights and biases are modified in response to network inputs only. There are no target outputs available. Most of these algorithms perform clustering operations. They categorize the input patterns into a finite number of classes. This is especially useful in such applications as vector quantization.

A cost function, also called performance function, is necessarily defined which is optimized by the adaptation of the weight values. The weights can be adjusted in accordance with the steepest gradient descent method.

$$W_{j+1} = W_j + \beta \Delta W_j \tag{2.1}$$

where $\Delta W_j = -\eta_j \frac{\partial Q(W_j)}{\partial W} + \Delta W_{j-1}$ $j = 0, 1, \cdots$, η_j and β are the learning rate and the momentum coefficient respectively. The momentum term is considered to prevent any local minimum of the performance function which traps the optimization path. Either slower convergence or oscillation of the performance value will often happen when a fixed, smaller leaning rate or larger one is chosen respectively. Adaptive learning rate is thus adopted to optimize the rate of convergence of the performance function. Generally speaking, the change of the learning rate depends on the change of the current performance value and the previous one. It goes up when the performance function of the current iteration is smaller than the last iteration one. The situation is reversed when the current performance value is larger than the previous one. In the meantime, the momentum term is ignored until the performance function is pulled down again.

From equation (2.1), the weights of $j+1$ iteration are updated by the sum of the change of weights at previous iteration and the gradient of the cost function with respect to the weights at j iteration. Most simply, we define a sum square error function as the cost function. That is,

$$Q(W) = \frac{1}{2}\Sigma_{i \in C} e_i^2(W) \tag{2.2}$$

where $e_i(W)$ is the error function of equation (2.2). The error function is defined differently in supervised and unsupervised manners. For supervised learning, with the targets information known, the error function is

$$e_i = t_i - p_i \tag{2.3}$$

On the other hand, the error function for unsupervised learning is represented as

$$e_i(W) = \mathbf{F}_i(W) \tag{2.4}$$

As shown in Figure 2.1, we are trying to minimize the error function $\mathbf{F}(W)$ as much as

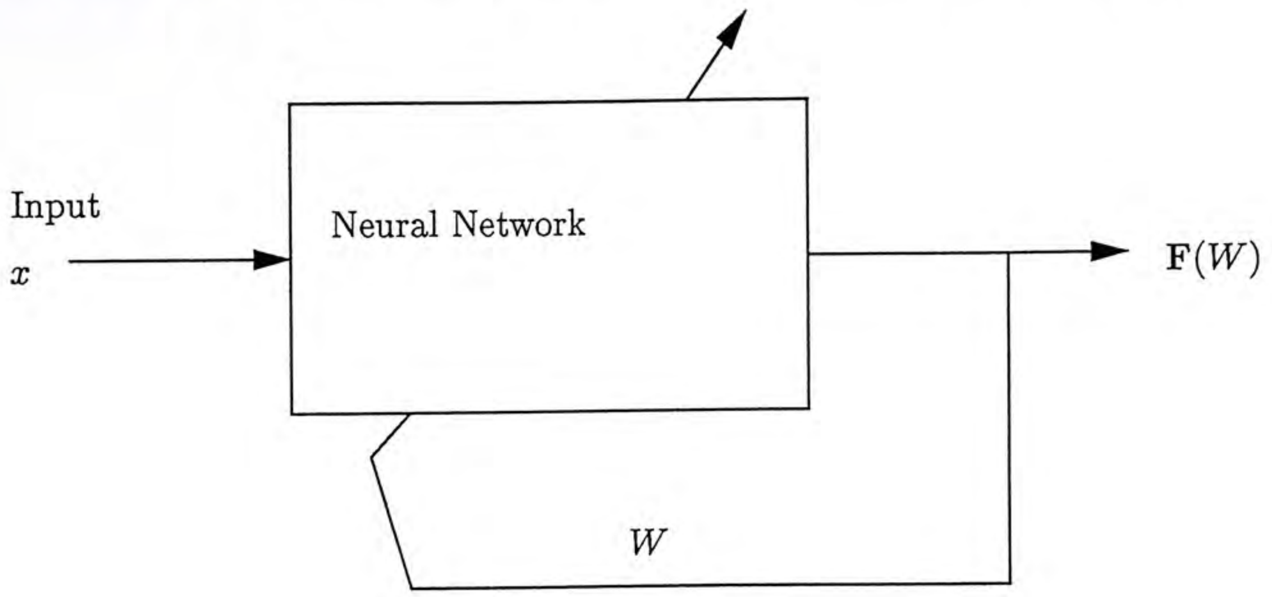


Figure 2.1: Block Diagram of Unsupervised Approximation

possible.

2.2 Example in Supervised Learning

As mentioned in the introduction, it is possible for approximating any nonlinear functions with supervised learning in case the input patterns and target patterns are available. Here, we demonstrate a nonlinear feedback control of a flexible-joint robot system. A state feedback control law for this system is designed using input-output linearization method which results in a stable closed-loop system. Our objective is to approximate the state feedback control law and the robot system so that the closed-loop neural controller/robot system behaves close to the desired closed-loop system up to a small error.

2.2.1 Problem Description

Shown in Figure 2.2 is a flexible-joint robot system that consists of a link driven by a motor through a torsional spring in the vertical plane. The equations of motion can be easily derived as [47]

$$I\ddot{q}_1 + MgLsinq_1 + k(q_1 - q_2) = 0 \tag{2.5}$$

$$J\ddot{q}_2 - k(q_1 - q_2) = u \tag{2.6}$$

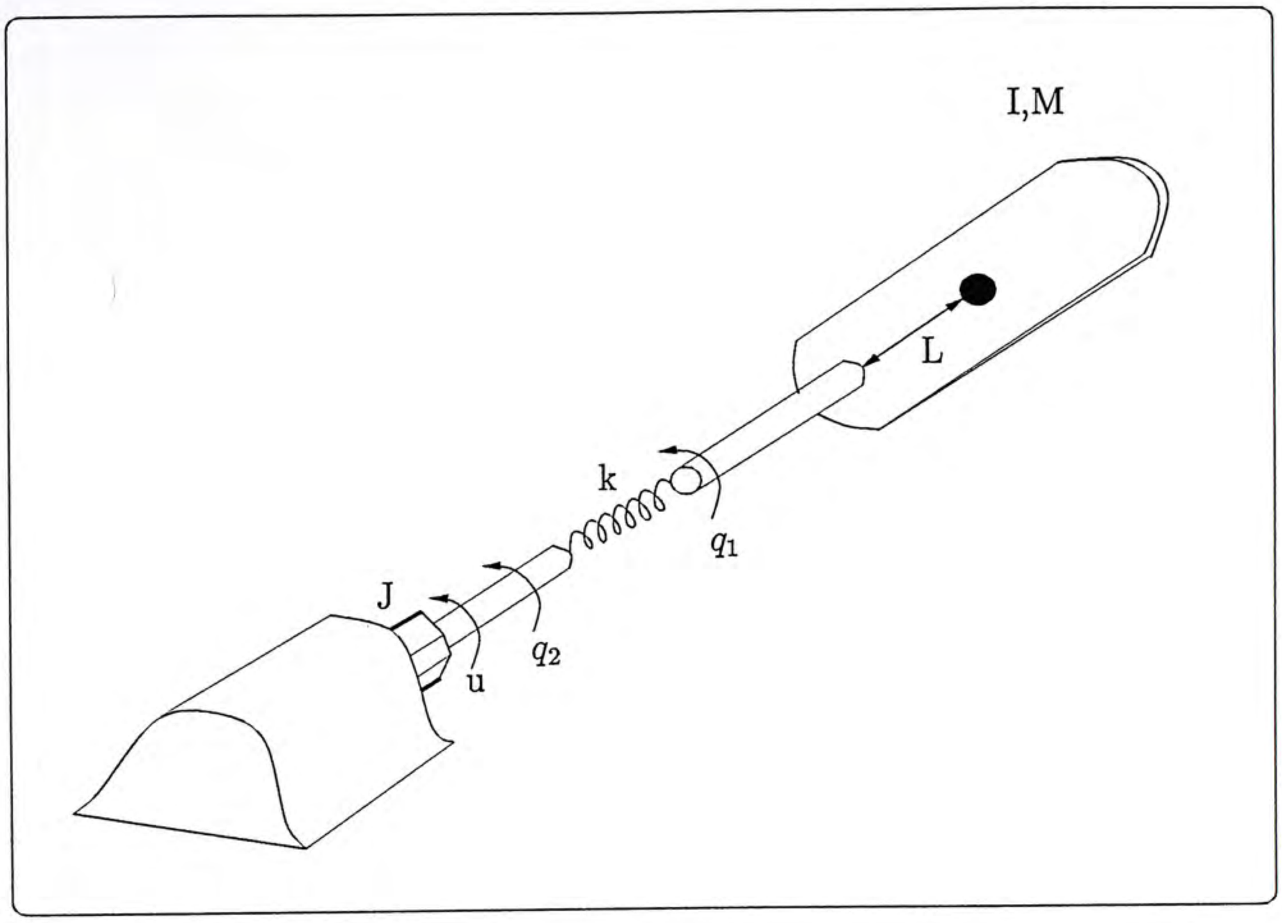


Figure 2.2: A Flexible-joint Robot System

The state-space representation of the system is

$$x = \begin{bmatrix} q_1 \\ \dot{q}_1 \\ q_2 \\ \dot{q}_2 \end{bmatrix}, \; f = \begin{bmatrix} x_2 \\ -\frac{MgL}{I} sinx_1 - \frac{k}{I}(x_1 - x_3) \\ x_4 \\ \frac{k}{j}(x_1 - x_3) \end{bmatrix}, \; g = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{1}{J} \end{bmatrix} \tag{2.7}$$

$$y = x_1 \tag{2.8}$$

Therefore, using input-output linearization method, the relative degree of the system can be defined by deriving the derivatives of the output y. That is,

$$\begin{aligned}
\dot{y} &= x_2 && (2.9)\\
\ddot{y} &= -\frac{MgL}{I} sinx_1 - \frac{k}{I}(x_1 - x_3) && (2.10)\\
y^{(3)} &= -\frac{MgL}{I} x_2 cosx_1 - \frac{k}{I}(x_2 - x_4) && (2.11)\\
y^{(4)} &= \frac{MgL}{I} sinx_1(x_2^2 + \frac{MgL}{I} cosx_1 + \frac{k}{I}) + \frac{k}{I}(x_1 - x_3)(\frac{k}{I} \\
&\quad frackJ + \frac{MgL}{I} cosx_1) + \frac{k}{IJ} u && (2.12)\\
&= \alpha(x) + \beta(x) u && (2.13)
\end{aligned}$$

where

$$\alpha(x) = \frac{MgL}{I} sinx_1(x_2^2 + \frac{MgL}{I}cosx_1 + \frac{k}{I}) + \frac{k}{I}(x_1 - x_3)(\frac{k}{I} + \frac{k}{J} + \frac{MgL}{I}cosx_1) \quad (2.14)$$

$$\beta(x) = \frac{k}{IJ} \quad (2.15)$$

Obviously, the relative degree is 4 and it is thus possible for designing the control law as follows:

$$\begin{aligned} u &= \frac{y^{(4)} - \alpha(x)}{\beta(x)} \\ &= \frac{y_d^{(4)} - \sum_{i=1}^{4} c_i e^{(4-i)} - \alpha(x)}{\beta(x)} \end{aligned} \quad (2.16)$$

where y_d is the reference input, $e^{(4-i)} = y^{(4-i)} - y_d^{(4-i)}$ are the errors between the actual output and the reference input as well as their derivatives. Under the control law (2.16), the tracking error e satisfies

$$e^{(4)} + c_1 e^{(3)} + c_2 \ddot{e} + c_3 \dot{e} + c_4 e = 0 \quad (2.17)$$

Therefore, if we choose the coefficients c_i's such that the polynomial $s^4 + c_1 s^3 + c_2 s^2 + c_3 s + c_4$ is Hurwitz, then we have $\lim_{t \to \infty} y = 0$. Here, we have chosen the roots at $[-2\ -2\ -2\ -2]$ so that the coefficients c_i will be [8 24 36 16].

2.2.2 Neural Network Configuration and Training

The control law given in (2.16) is highly nonlinear and complex. We will consider to approximate this control law by feedforward neural networks. Figure 2.3 shows a system which comprises a neural controller and the actual robot system. However, it is impracticable to approximate the desired linearized system by this system since it is difficult to train the neural controller throughout. To this end, we need to create a neural robot model in order to train the neural controller. Therefore, we need to use two, as shown in the introduction, three layers feedforward neural networks to approximate the nonlinear robot system and the state feedback controller. Since supervised approximation is adopted, it is necessary to define all the input patterns and target patterns. The input patterns are chosen with all possible combination of the robot movement and the steady state. The initial condition of the system is also considered.

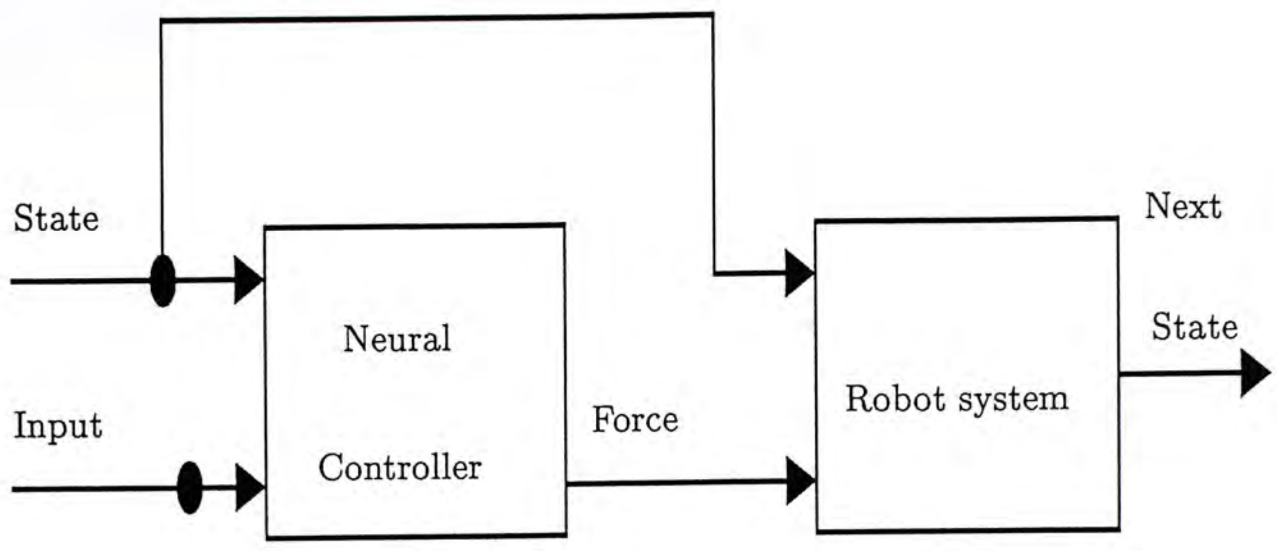


Figure 2.3: Neural Controller/Robot System

The states of the robot system and the linearized system are obtained by simulating the system for 0.05 sec using MATLAB function "ODE23" while the initial patterns put in this function are the input patterns. The target patterns make use of the difference between the next state and the input state. The input and target patterns of the linearized system are used for the approximation of the state feedback controller. We will find a neural network controller which takes the current robot arm angle, velocity and the inputs, and gives out a current value which can be applied to the robot system. The current value should make the next state of the robot system identical to that defined by the desired linearized system. Therefore, the neural robot model should have two outputs representing the angle and the velocity while the neural controller model has one output only.

The neural network models are learned to predict the change in state over 0.05 sec because the state does not change by a large amount in this period of time, and we can improve the performance of the model if we predict only the change in state. If we need to know the actual state we simply need to add the change to the previous state.

The neural robot model is a tansig/purelin network with 8 hidden neurons. The outputs of the network are the angle and the velocity of the robot system. The results of the approximation of the robot system are shown below.

2.2.3 Simulation Result

Having obtained the neural robot system whose performance is quite close to that of the actual robot system, we can make use of it in training the neural controller. Similarly,

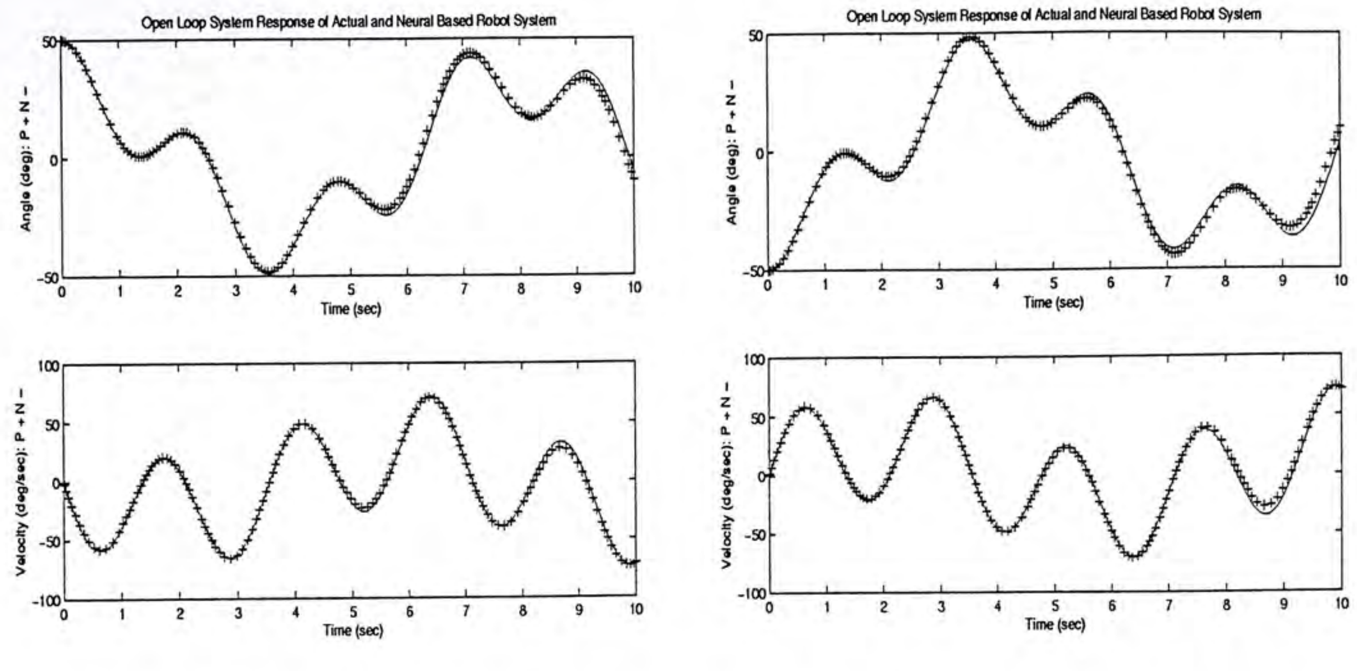


Figure 2.4: Simulation Result of the Neural Robot System at $x_1 = \pm 50$ degrees

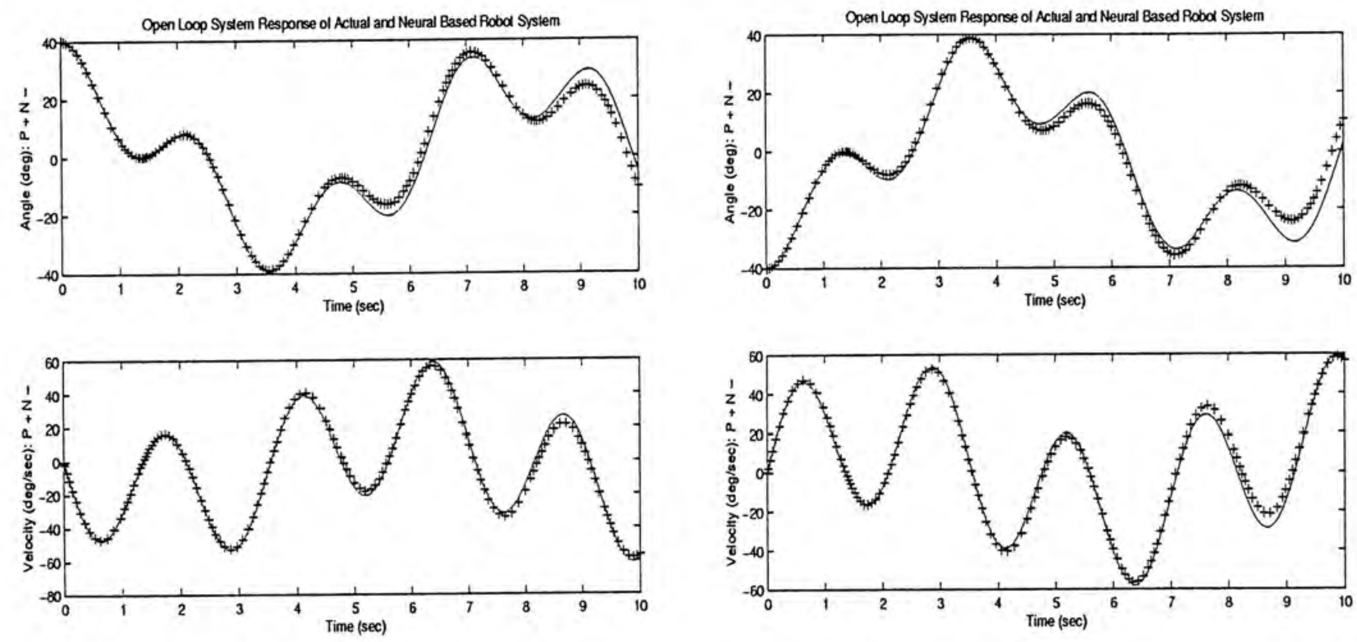


Figure 2.5: Simulation Result of the Neural Robot System at $x_1 = \pm 40$ degrees

the neural controller use same network with only one force output. As the neural robot model has been trained with performance error less than $1e^{-5}$, a neural controller-neural robot model network is thus generated by combining up these two models. As shown in Figure 2.9, the error occurs at the output of the neural robot model. The derivatives of this error can be back-propagated through the neural robot model to the neural controller. The error are then back-propagated through the neural controller and used to adjust its weights and biases while the values of the weights and biases of the neural robot model are kept *unchanged.* Thus the control network must learn how to control the robot system so that it behaves like the linearized system. After the training process is completed with

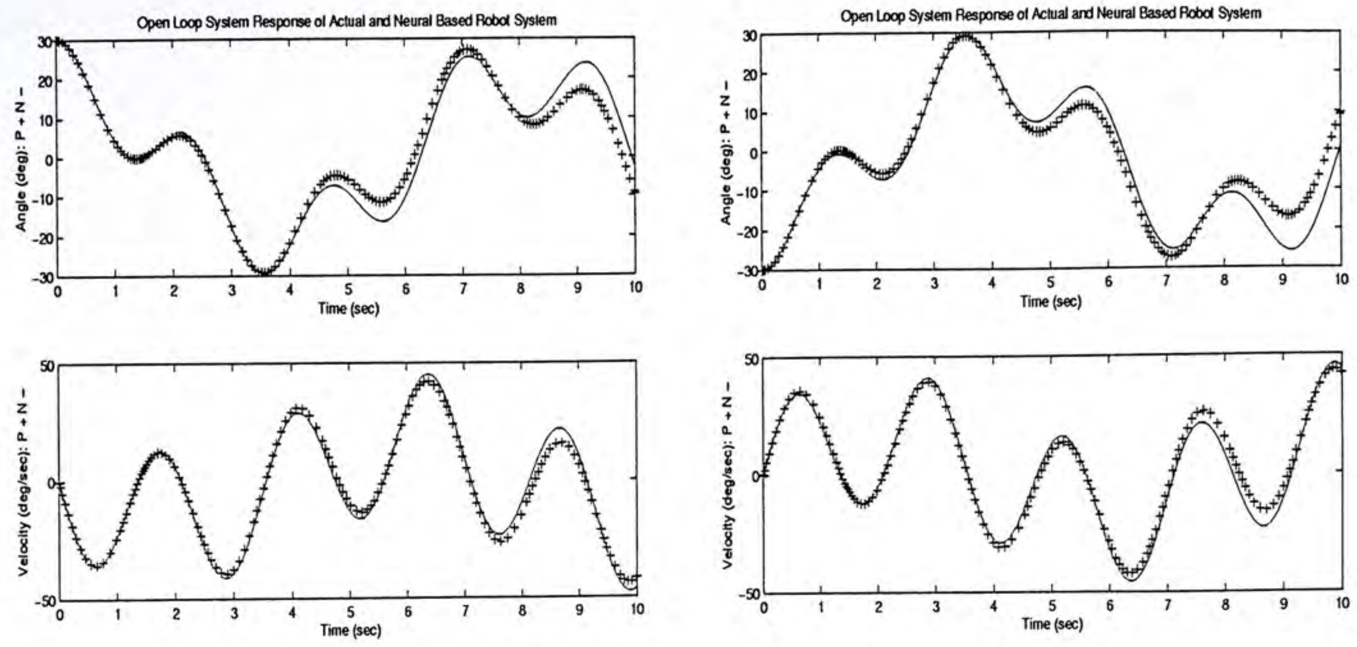


Figure 2.6: Simulation Result of the Neural Robot System at $x_1 = \pm 30$ degrees

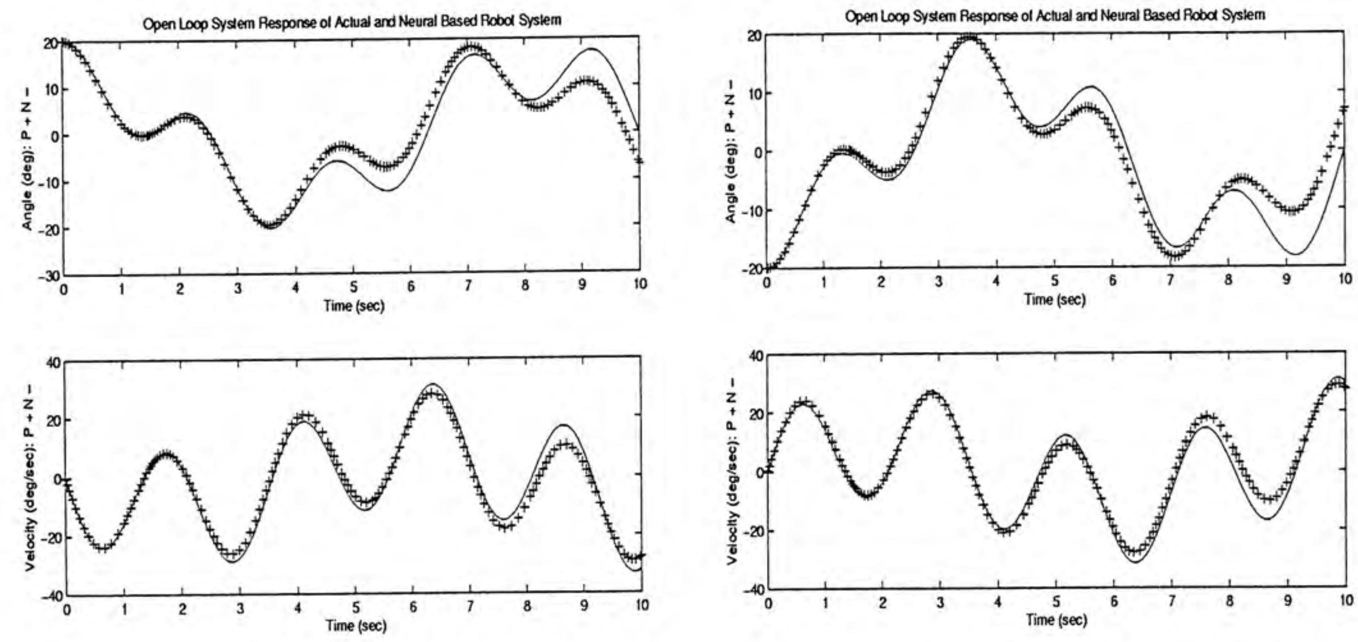


Figure 2.7: Simulation Result of the Neural Robot System at $x_1 = \pm 20$ degrees

performance error less than $1e^{-3}$, we have to test the neural controller-robot system.

We have simulated the neural controller-robot system with initial angle of 10 degrees with 0 velocity, and two desired constant input angle of 60, -60, 40, -40, 20, 20 degrees. The network does a near perfect job of making the nonlinear robot system act like the linearized system even for a wider range of degree of movement of the robot arm.

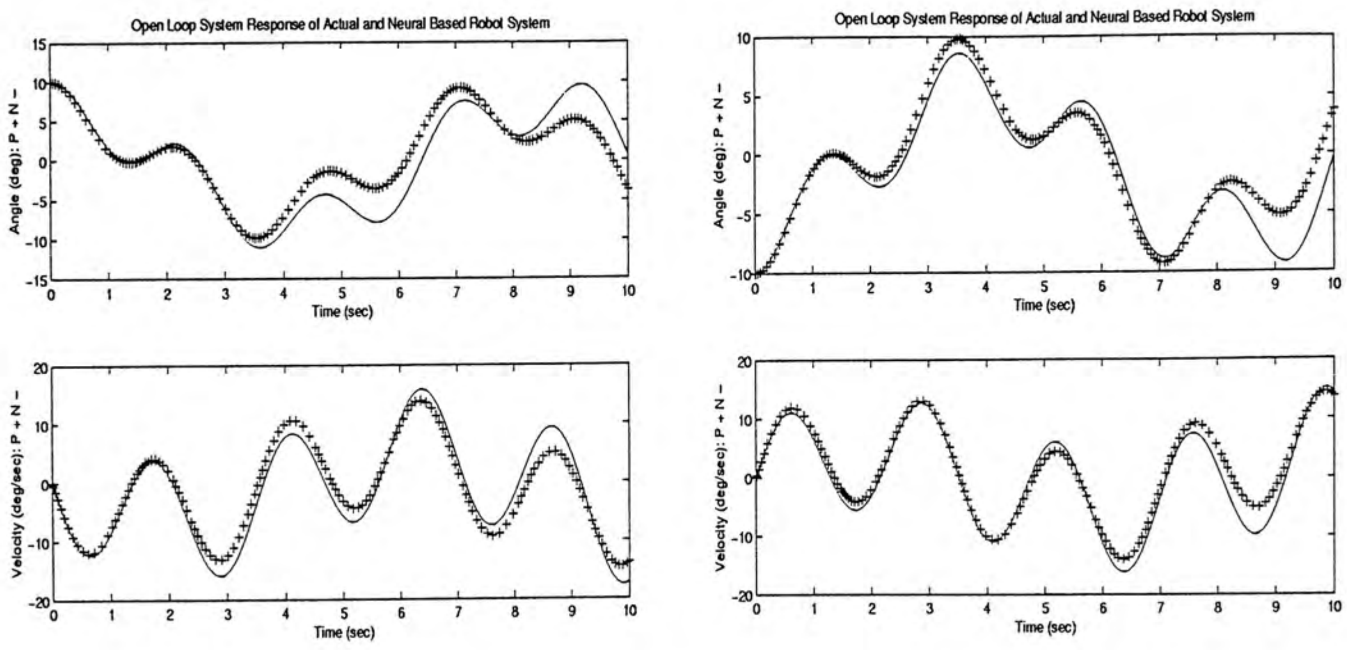


Figure 2.8: Simulation Result of the Neural Robot System at $x_1 = \pm 10$ degrees

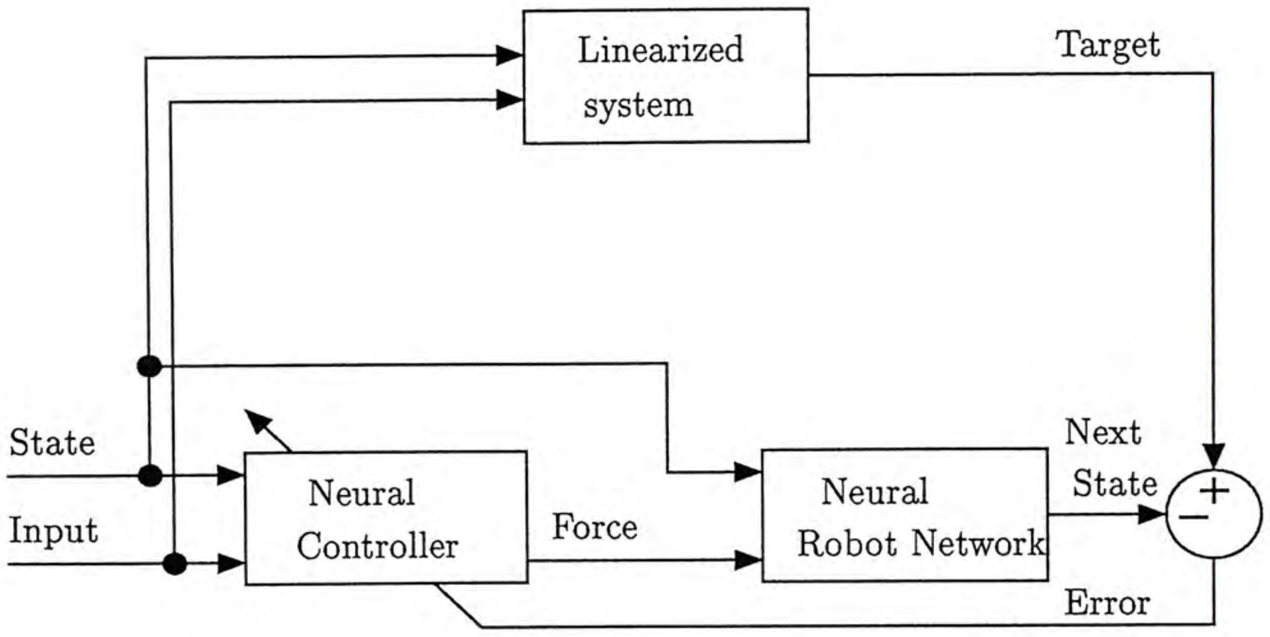


Figure 2.9: Neural Controller/Neural Robot Network

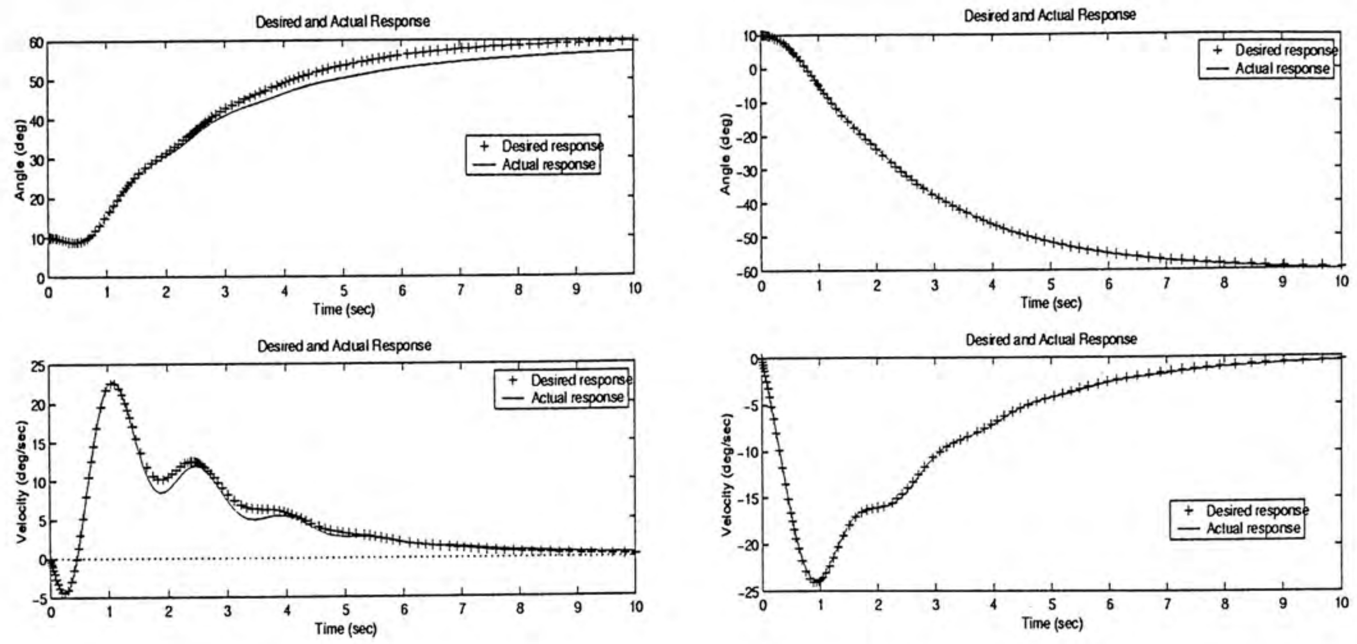


Figure 2.10: Simulation Result with Desired Input Angle ±60 degrees

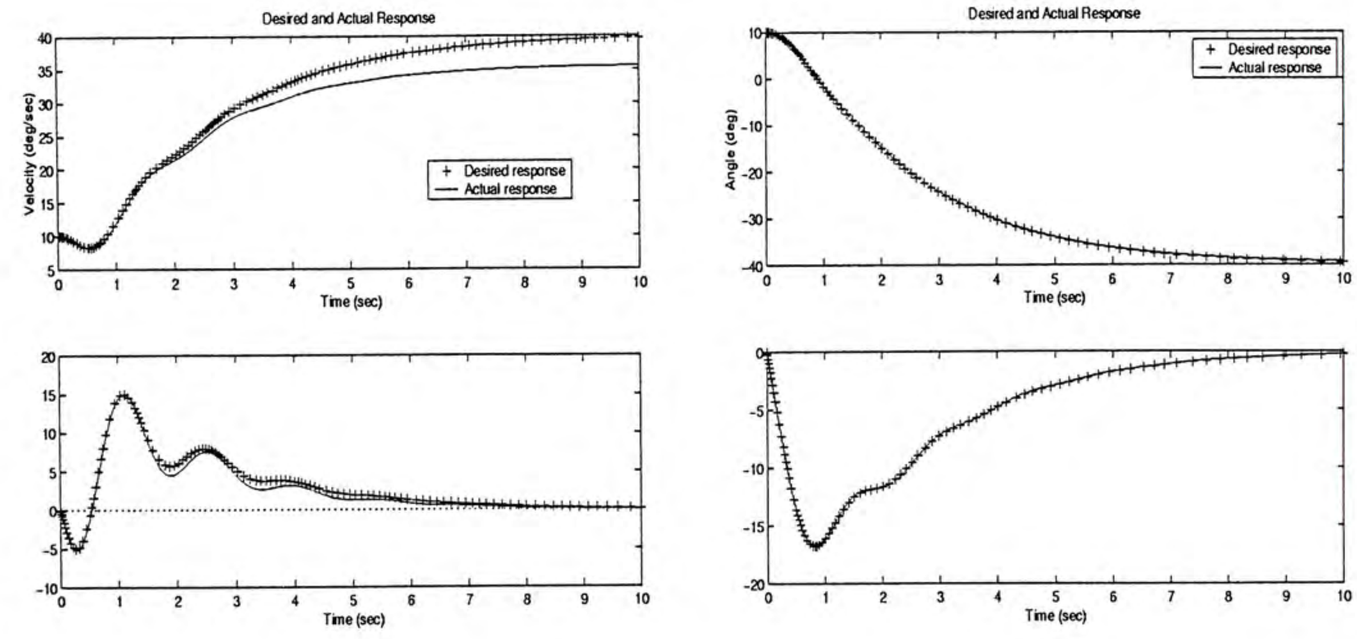


Figure 2.11: Simulation Result with Desired Input Angle ±40 degrees

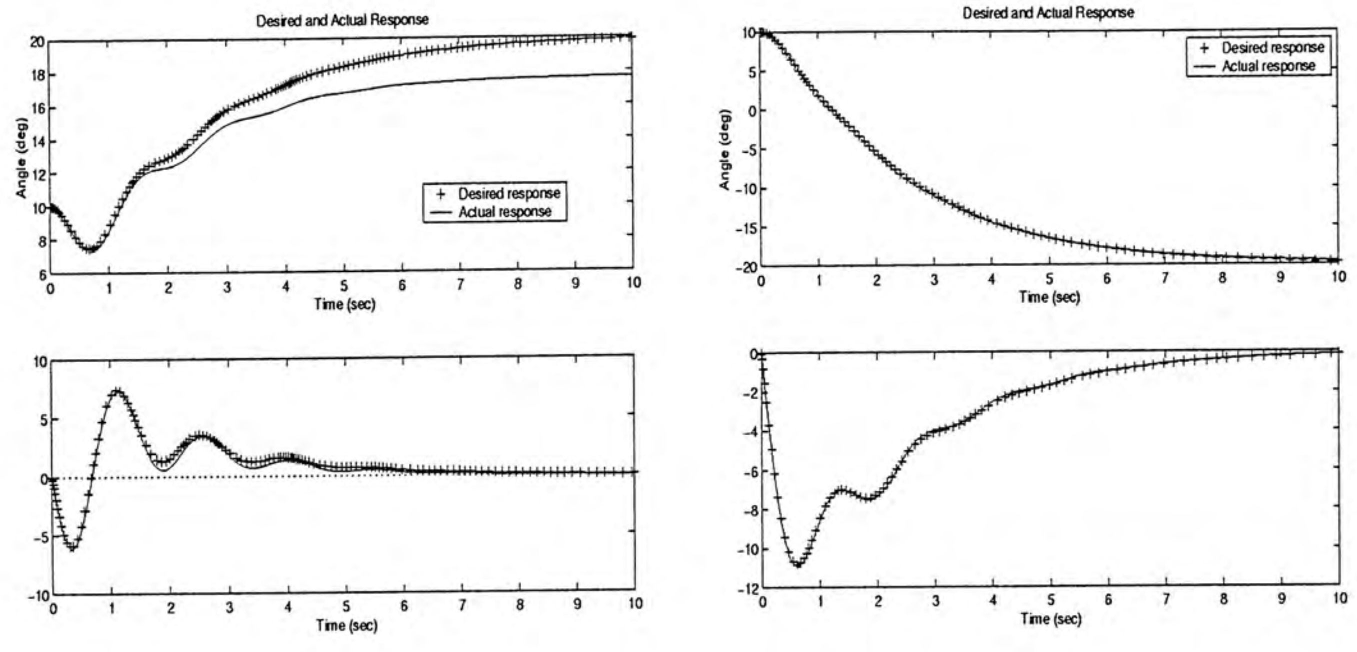


Figure 2.12: Simulation Result with Desired Input Angle ±20 degrees

Chapter 3

Neural Based Approximation of Center Manifold Equations

3.1 Solving Center Manifold Equations by Feedforward Neural Network Approximation

In this Chapter, we will consider to solve a nonlinear partial differential equation of the following form

$$\frac{\partial \mathbf{z}(v)}{\partial v} S v = \alpha(\mathbf{z}(v), v) \tag{3.1}$$

where $v \in R^q$, $S \in R^{q \times q}$, and $\alpha(\cdot, \cdot) : R^n \times R^q \to R^n$ is a sufficiently smooth function defined in a neighborhood of the origin $R^n \times R^q \to R^n$ satisfying $\alpha(0,0) = 0$. It is assumed that all the eigenvalues of S have zero real parts, and all the eigenvalues of $\frac{\partial \alpha}{\partial z}(0,0)$ have nonzero real parts. The local solution $\mathbf{z}(v)$ of (3.1) characterizes the center manifold of the following system [31]:

$$\begin{aligned} \dot{z} &= \alpha(z, v) \\ \dot{v} &= S v \end{aligned} \tag{3.2}$$

Equation (3.1) is called the center manifold equations for (3.2). Equation (3.2) is the special case of (A.6) and (A.7) in Appendix A with g_2 equals to zero. The center manifold equations arise in many control problems such as the nonlinear output regulation problem which aims to design a control law for a nonlinear plant to achieve asymptotic tracking and disturbance rejection in the closed-loop system.

Due to the nonlinear nature of the center manifold equations (3.1), it is usually impossible to obtain an exact solution. The way for solving this kind of problem is to develop an approximation method. The feedforward neural network can approximately solve such equation according to the universal approximation theorem. Hence, in this chapter, we are going to develop a neural based approach to solve the center manifold equations approximately. As a result of the Universal Approximation Theorem, given any $\gamma > 0$, there exist an integer N, scalars $w_{1i}^O, \cdots, w_{ni}^O, w_{10}^O, \cdots, w_{n0}^O, w_{i1}^I, \cdots, w_{in}^I$, and w_{i0}^I such that

$$\hat{\mathbf{z}}(v) = \begin{bmatrix} \sum_{i=1}^{N} w_{1i}^O \phi(w_{i1}^I v + w_{i0}^I) + w_{10}^O \\ \vdots \\ \sum_{j=1}^{N} w_{ni}^O \phi(w_{in}^I v + w_{i0}^I) + w_{n0}^O \end{bmatrix} \tag{3.3}$$

satisfies

$$\begin{aligned} &\max_{v\in\Gamma} ||\mathbf{z}(v) - \hat{\mathbf{z}}(v)|| < \gamma, \\ &\max_{v\in\Gamma} ||\tfrac{\partial(\mathbf{z}(v)-\hat{\mathbf{z}}(v))}{\partial v}|| < \gamma \end{aligned} \tag{3.4}$$

Moreover, using the technique developed in [29], it is possible to show that the center manifold equations can be solved up to an arbitrarily small error ϵ_r, that is,

$$||\frac{\partial \hat{\mathbf{z}}(v)}{\partial v} Sv - \alpha(\hat{\mathbf{z}}(v), v)|| < \epsilon_r, \quad v \in V \tag{3.5}$$

Next, we will consider the problem of how to find an desirable neural network which satisfies (3.5). For clarity, we will use the notations $\hat{\mathbf{z}}(W, v)$ to explicitly indicate the reliance of $\hat{\mathbf{z}}$ on the weight vector W which consists of all weights as given in (3.3). The dimension of W is denoted by s_N which is determined by the number N of the hidden neurons. Let

$$J(W, v) = \frac{1}{2} ||\frac{\partial \hat{\mathbf{z}}(W, v)}{\partial v} Sv - \alpha(\hat{\mathbf{z}}(W, v), v)||^2 \tag{3.6}$$

Clearly, if for some N and $W \in R^{s_N}$,

$$J(W, v) < \epsilon_r^2, \quad \forall v \in \Gamma \tag{3.7}$$

Then $\hat{\mathbf{z}}(W, v)$ satisfies (3.5). Nevertheless, it is difficult to solve (3.7) since $J(W, v)$ depends on both W and v. Therefore we will discretize (3.7) by defining

$$Q(W) = \sum_{v\in\Gamma_d} J(W, v) \tag{3.8}$$

where Γ_d is a subset of Γ consisting of finitely many elements of Γ. When Γ_d is sufficiently *dense* in Γ, then $Q(W) \leq \epsilon_r^2$ will lead to a good approximation of the solution of (3.5).

Since, for each fixed N, $Q(W)$ relies only on the parameter W, the optimal weight that minimizes $Q(W)$ can be searched by any minimization technique which has been mentioned in chapter 2. For example, gradient mehtod gives

$$W_{j+1} = W_j + \beta \Delta W_j \tag{3.9}$$

Thus the problem of looking for the approximated solution of the center manifold equations is converted into a parameter optimization problem.

3.2 Example

3.2.1 Problem Description

Here we introduce an example to demonstrate the efficiency of our approach described in the last section. We will show how to solve the center manifold equations approximately. Consider the center manifold equations shown as follows:

$$\begin{aligned} \frac{\partial \mathbf{x}_1(v)}{\partial v_1} a v_2 - \frac{\partial \mathbf{x}_1(v)}{\partial v_2} a v_1 &= -\mathbf{x}_1(v) + v_1 && (3.10) \\ \frac{\partial \mathbf{x}_2(v)}{\partial v_1} a v_2 - \frac{\partial \mathbf{x}_2(v)}{\partial v_2} a v_1 &= -\mathbf{x}_2(v) + \mathbf{x}_1(v) v_1 && (3.11) \end{aligned}$$

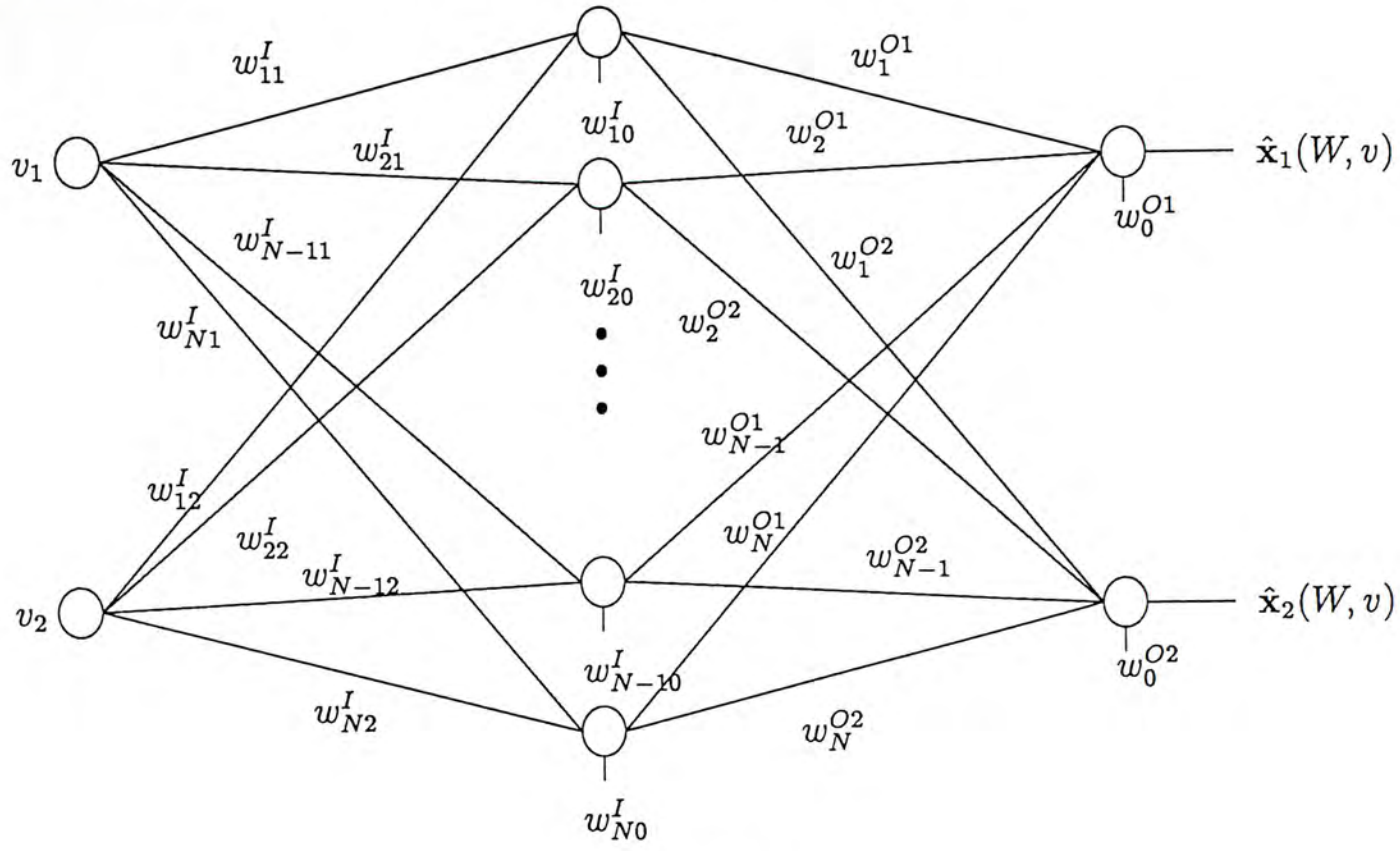
where $a > 0$. The above two equations can be put into the form of (3.1) as follows with

$$\mathbf{z}(v) = \begin{bmatrix} \mathbf{x}_1(v) \\ \mathbf{x}_2(v) \end{bmatrix}, \ \alpha(\mathbf{z}, v) = \begin{bmatrix} -\mathbf{x}_1(v) + v_1 \\ -\mathbf{x}_2(v) + \mathbf{x}_1(v) v_1 \end{bmatrix}, \ S = \begin{bmatrix} 0 & a \\ -a & 0 \end{bmatrix}$$

The partial differential equations (3.10) and (3.11) can be viewed as the center manifold equations for the following system

$$\begin{aligned} \dot{x}_1 &= -x_1 + v_1 \\ \dot{x}_2 &= -x_2 + x_1 v_1 \\ \dot{v}_1 &= a v_2 \\ \dot{v}_2 &= a v_1 \end{aligned}$$

Our goal is to solve the above center manifold equations (3.10) and (3.11) approximately to a prescribed small error by using neural based method. We then construct a feedforward neural network in which the outputs of the neural model are the approximated solution of the unknown functions $\mathbf{x}_1(v)$ and $\mathbf{x}_2(v)$. Figure 3.1 shows the approximated solutions



Hidden Layer

Figure 3.1: A Neural Model for Solving the Center Manifold Equations

$\hat{\mathbf{x}}_1(W,v)$ and $\hat{\mathbf{x}}_2(W,v)$ which approximate the solution of the center manifold equations (3.10) and (3.11). The approximated solution is of the form

$$\hat{\mathbf{x}}_1(W,v) = \sum_{i=1}^{N} w_i^{O1}\phi(M_i) + w_0^{O1} \tag{3.12}$$

$$\hat{\mathbf{x}}_2(W,v) = \sum_{i=1}^{N} w_i^{O2}\phi(M_i) + w_0^{O2} \tag{3.13}$$

where

$$M_i = \sum_{j=1}^{2} w_{ij}^I v_i + w_{i0}^I \tag{3.14}$$

and $\phi(.)$ is the hypertangent activation function used in the hidden neurons. That is

$$\phi(M_i) = \frac{e^{M_i} - e^{-M_i}}{e^{M_i} + e^{-M_i}} \tag{3.15}$$

and its derivative is

$$\phi'(M_i) = 1 - \phi^2(M_i) \tag{3.16}$$

In order to find the approximated solution of the center manifold equations, let

$$\begin{aligned}
e_1 &= \frac{\partial \hat{\mathbf{x}}_1(W,v)}{\partial v} Sv + \hat{\mathbf{x}}_1(W,v) - v_1 && (3.17)\\
e_2 &= \frac{\partial \hat{\mathbf{x}}_2(W,v)}{\partial v} Sv + \hat{\mathbf{x}}_2(W,v) - \hat{\mathbf{x}}_1(W,v)v_1 && (3.18)\\
J(W,v) &= \frac{1}{2}(e_1^2 + e_2^2) && (3.19)\\
Q(W) &= \sum_{v\in\Gamma_d} J(W,v) && (3.20)
\end{aligned}$$

The derivatives of $Q(W)$ with respect to weights and biases are derived as follows:

$$\begin{aligned}
\frac{\partial Q(W)}{\partial w_i^{O1}} &= \sum_{p=1}^{T_p}[e_{1p}(\frac{\partial e_{1p}}{\partial w_i^{O1}}) + e_{2p}(\frac{\partial e_{2p}}{\partial w_i^{O1}})] && (3.21)\\
\frac{\partial Q(W)}{\partial w_i^{O2}} &= \sum_{p=1}^{T_p}[e_{1p}(\frac{\partial e_{1p}}{\partial w_i^{O2}}) + e_{2p}(\frac{\partial e_{2p}}{\partial w_{ij}^{O2}})] && (3.22)\\
\frac{\partial Q(W)}{\partial w_{ij}^{I}} &= \sum_{p=1}^{T_p}[e_{1p}(\frac{\partial e_{1p}}{\partial w_{ij}^{I}}) + e_{2p}(\frac{\partial e_{2p}}{\partial w_{ij}^{I}})] && (3.23)\\
\frac{\partial Q(W)}{\partial w_0^{O1}} &= \sum_{p=1}^{T_p}[e_{1p}(\frac{\partial e_{1p}}{\partial w_0^{O1}}) + e_{2p}(\frac{\partial e_{2p}}{\partial w_0^{O1}})] && (3.24)\\
\frac{\partial Q(W)}{\partial w_0^{O2}} &= \sum_{p=1}^{T_p}[e_{1p}(\frac{\partial e_{1p}}{\partial w_0^{O2}}) + e_{2p}(\frac{\partial e_{2p}}{\partial w_0^{O2}})] && (3.25)\\
\frac{\partial Q(W)}{\partial w_{i0}^{I}} &= \sum_{p=1}^{T_p}[e_{1p}(\frac{\partial e_{1p}}{\partial w_{i0}^{I}}) + e_{2p}(\frac{\partial e_{2p}}{\partial w_{i0}^{I}})] && (3.26)
\end{aligned}$$

where

$$\begin{aligned}
\frac{\partial e_{1p}}{\partial w_i^{O1}} &= \phi'(M_i)(w_{i1}^I a v_2 - w_{i2}^I a v_1) + \phi(M_i) && (3.27)\\
\frac{\partial e_{2p}}{\partial w_i^{O1}} &= -\phi(M_i)v_1 && (3.28)\\
\frac{\partial e_{1p}}{\partial w_i^{O2}} &= 0 && (3.29)\\
\frac{\partial e_{2p}}{\partial w_i^{O2}} &= \phi'(M_i)(w_{i1}^I a v_2 - w_{i2}^I a v_1) + \phi(M_i) && (3.30)\\
\frac{\partial e_{1p}}{\partial w_{ij}^{I}} &= -2w_i^{O1}\phi(M_i)\phi'(M_i)(w_{i1}^I a v_2 - w_{i2}^I a v_1)v_j \\
&\quad +\phi'(M_i)w_i^{O1} a v_j + \phi'(M_i)w_i^{O1} v_j && (3.31)\\
\frac{\partial e_{2p}}{\partial w_{ij}^{I}} &= -2w_i^{O2}\phi(M_i)\phi'(M_i)(w_{i1}^I a v_2 - w_{i2}^I a v_1)v_j + \phi'(M_i)w_i^{O2} a v_j \\
&\quad +\phi'(M_i)w_i^{O2} v_j - \phi'(M_i)w_i^{O1} v_j v_1 && (3.32)\\
\frac{\partial e_{1p}}{\partial w_0^{O1}} &= 1 && (3.33)
\end{aligned}$$

$$\frac{\partial e_{2p}}{\partial w_0^{O1}} = -v_1 \quad (3.34)$$

$$\frac{\partial e_{1p}}{\partial w_0^{O2}} = 0 \quad (3.35)$$

$$\frac{\partial e_{2p}}{\partial w_0^{O2}} = 1 \quad (3.36)$$

$$\frac{\partial e_{1p}}{\partial w_{i0}^{I}} = -2w_i^{O1}\phi(M_i)\phi'(M_i)(w_{i1}^{I}av_2 - w_{i2}^{I}av_1) + \phi'(M_i)w_i^{O1} \quad (3.37)$$

$$\frac{\partial e_{2p}}{\partial w_{i0}^{I}} = -2w_i^{O2}\phi(M_i)\phi'(M_i)(w_{i1}^{I}av_2 - w_{i2}^{I}av_1) + \phi'(M_i)w_i^{O2} - \phi'(M_i)w_i^{O1}v_1 \quad (3.38)$$

With 15 hidden neurons and a set of training pattern in a defined region $||v|| \leq 2$, we have implemented our approximated approach to minimize $Q(W)$ up to 10^{-4}.

3.2.2 Simulation Result

The approximated solution will be compared with the exact solution $\mathbf{x}_1(v)$ and $\mathbf{x}_2(v)$:

$$\mathbf{x}_1(v) = \frac{1}{1+a^2}(v_1 - av_2) \quad (3.39)$$

$$\mathbf{x}_2(v) = \frac{1}{1+5a^2+4a^4}((1+a^2)v_1^2 - 3av_1v_2 + 3a^2v_2^2) \quad (3.40)$$

where $a = 3$ and the graphs shown below are obtained from our approach and the exact solution respectively.

Figures 3.2 and 3.3 show the simulation results from our neural based approximation and the exact solution. Obviously, the approximated solution is closed to the exact solution as shown in Figure 3.4.

Figures 3.5 and 3.7 show the partial derivatives of $\hat{\mathbf{x}}_1(W, v)$, $\mathbf{x}_1(v)$, $\hat{\mathbf{x}}_2(W, v)$ and $\mathbf{x}_2(v)$ respectively. We also found that the feedforward neural network approximation indeed work in high precision which has been indicated in Figure 3.7. We show that the approximated solution from our approach is close to the exact solution. Consequently, it implies that the neural based approximation method is an efficient way for solving the center manifold equations. On the other hand, during the training process, we have investigated some important factors affecting the approximation efficiency and accuracy.

3.2.3 Discussion

The solutions derived from the center manifold equation are linear function and quadratic function of v which is an odd and even function respectively. The existence of the bias

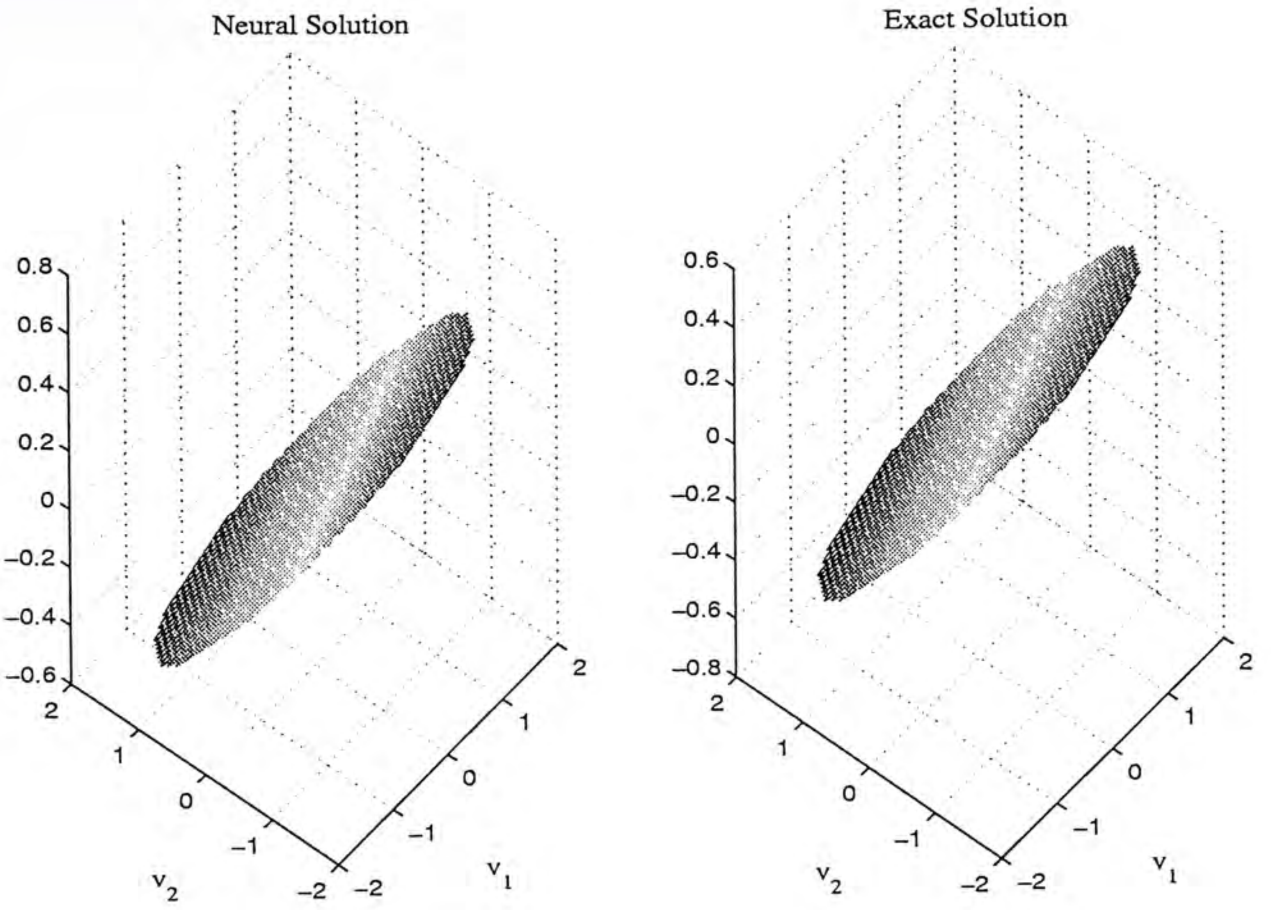


Figure 3.2: Solution of $\hat{\mathbf{x}}_1(W, v)$ and $\mathbf{x}_1(v)$ for $||v|| \leq 2$

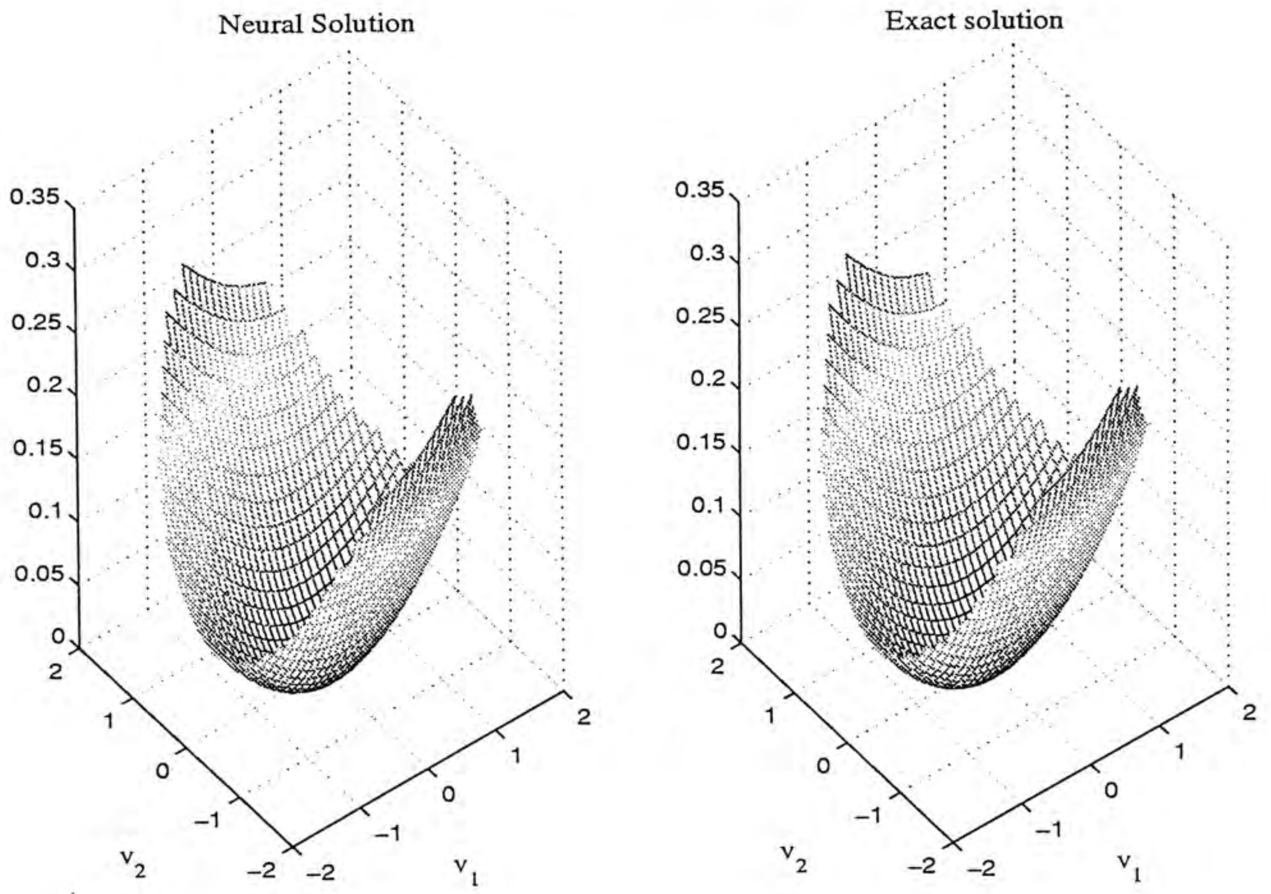


Figure 3.3: Solution of $\hat{\mathbf{x}}_2(W, v)$ and $\mathbf{x}_2(v)$ for $||v|| \leq 2$

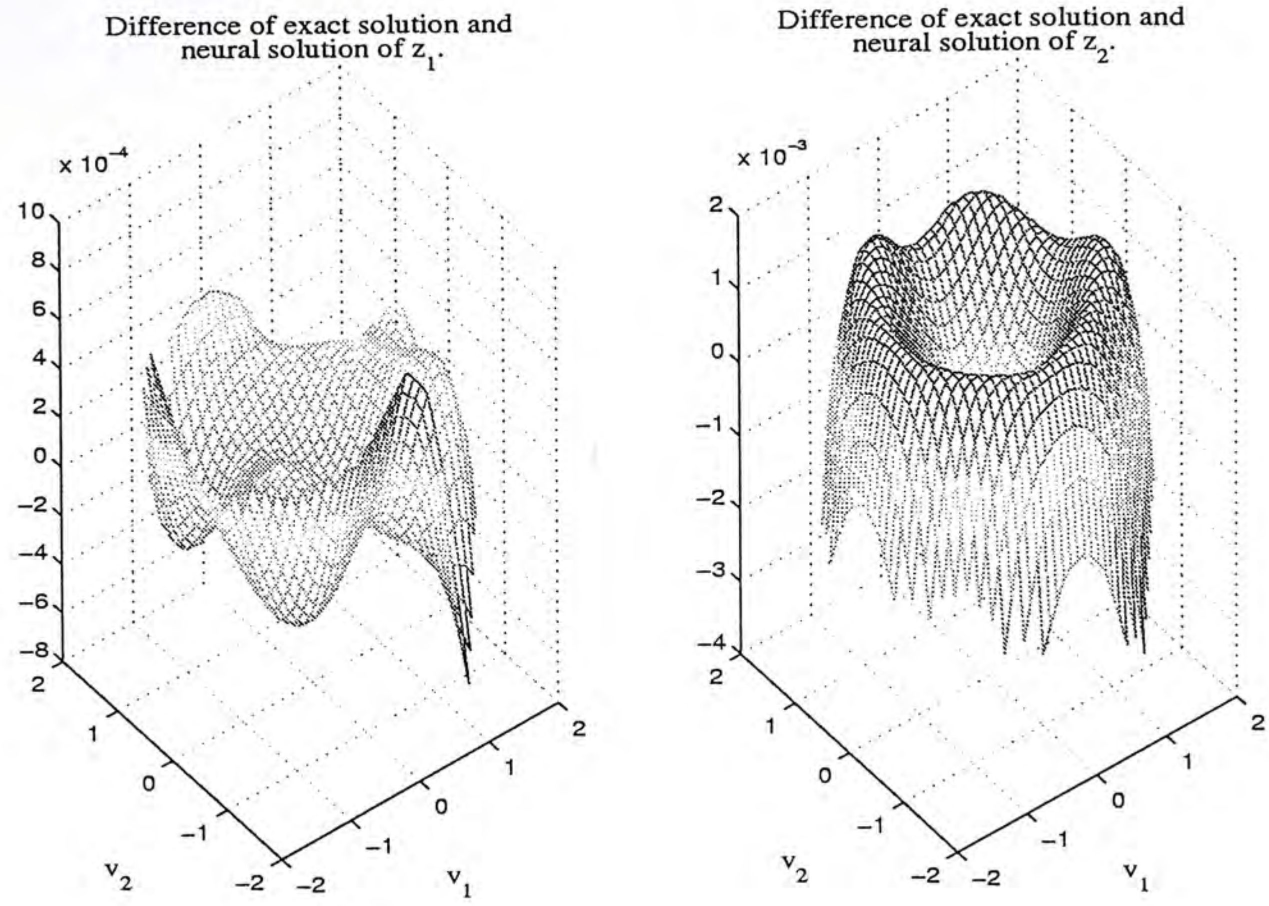


Figure 3.4: Difference Between the Exact Solution and the Neural Based Solution for $||v|| \leq 2$

term is a crucial factor for the approximation in this example. Without the bias term, we cannot get any satisfied approximated solution even the performance error is small enough since the hyperbolic tangent is odd and radial basis is even. However the bias term can be neglected if the functions to be approximated are known to be even or odd a priori.

The complexity of the system and the number of the training patterns used will also affect the optimization performance. In this example, since the system is simple, we have to handle the latter factor only. Using different number of hidden neurons of the network gives the same condition of the system, the time spent on minimizing the performance function to the desired value is greatly different. Below we show two couples of the training processes for this argument. The first training is done in a smaller region $||v|| \leq 1$ and the second training is implemented in a larger region $||v|| \leq 2$. The larger the region is, the more the training patterns are required.

Within a smaller region, it is sufficient to use fewer neurons, e.g. 8, to yield a fast rate of convergence in Table 3.1. The situation is opposite for the case in a larger region which is shown in Table 3.2. Now the problem of solving the partial differential equation from

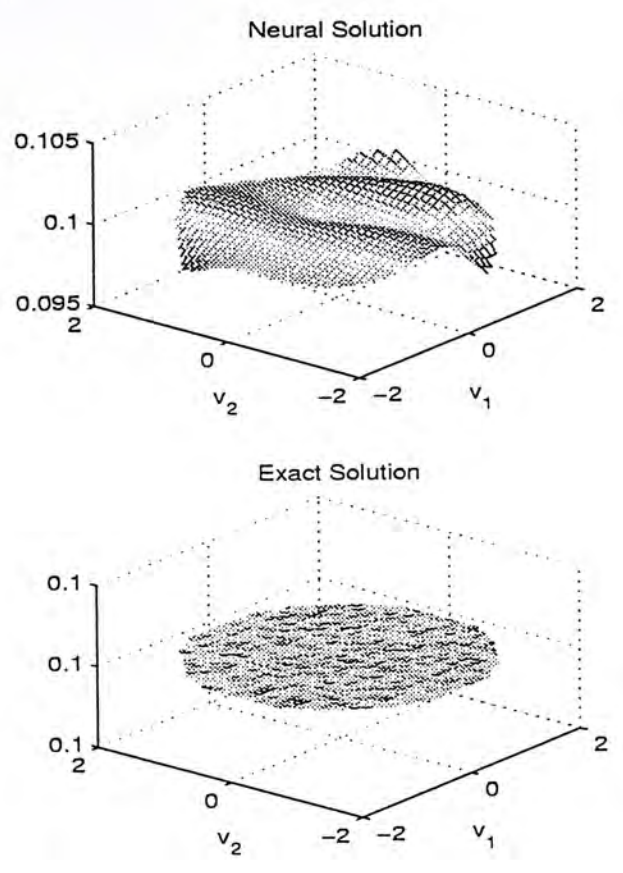


Figure 3.5: Partial Derivatives of $\hat{\mathbf{x}}_1(W, v)$ and $\mathbf{x}_1(v)$ for $||v|| \leq 2$

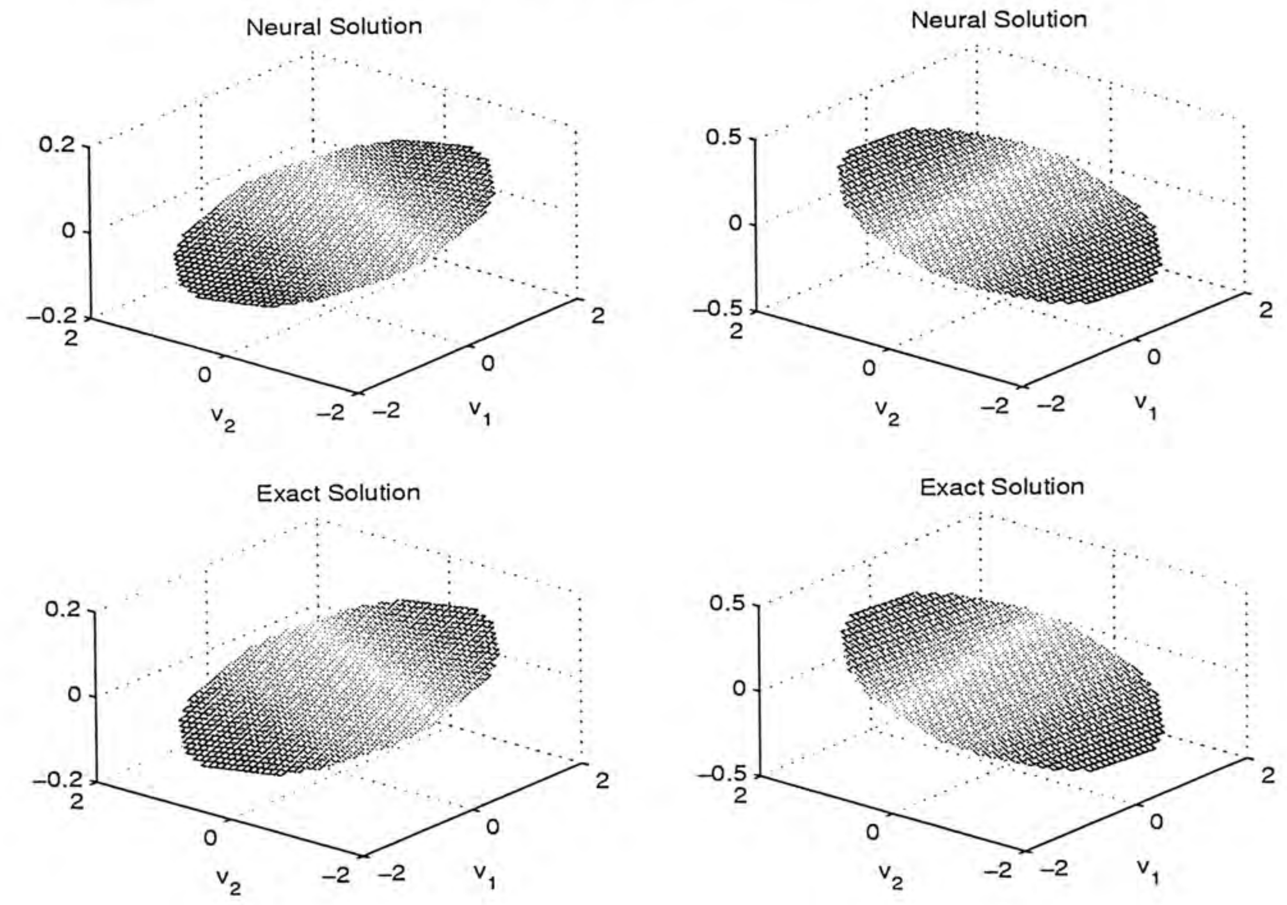


Figure 3.6: Partial Derivatives of $\hat{\mathbf{x}}_2(W, v)$ and $\mathbf{x}_2(v)$ for $||v|| \leq 2$

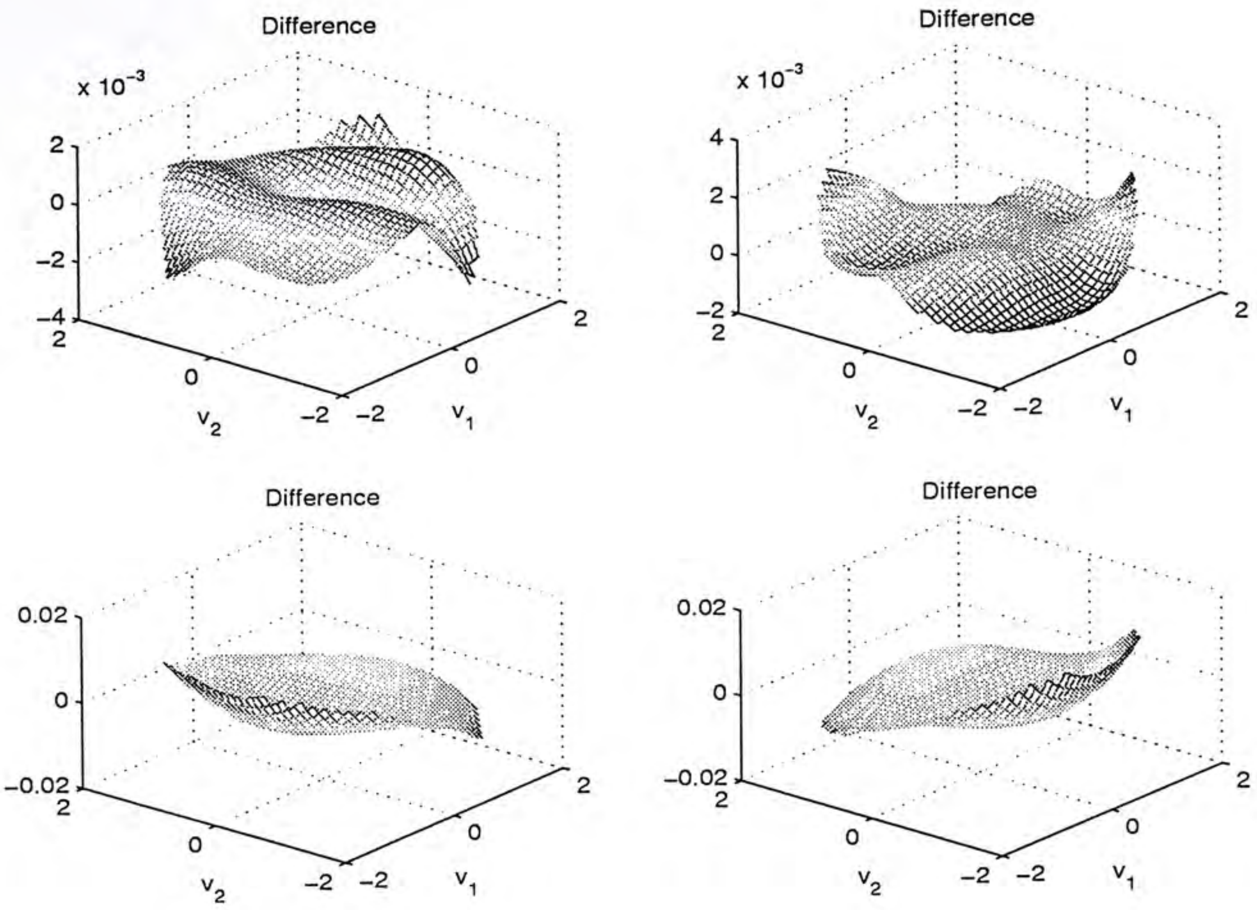


Figure 3.7: Difference of the Partial derivatives for $||v|| \leq 2$

Hidden neurons	$N = 8$	$N = 10$
No. of iterations	1.5×10^5	7×10^5

Table 3.1: Training Performance with Hidden Neurons Within the Region $||v|| \leq 1$

Hidden neurons	$N = 15$	$N = 20$
No. of iterations	29×10^5	12×10^5

Table 3.2: Training Performance with Hidden Neurons Within the Region $||v|| \leq 2$

numerical calculation is converted into parameters optimization problem, we can easily implement this method efficiently.

Chapter 4

Connection of Center Manifold Equations to Output Regulation Problem

4.1 Output Regulation Theory

The solution of the center manifold equations can be used to solve the output regulation problem proposed in [32], [28]. More precisely, the problem can be described as follows:

Consider a nonlinear plant which is described as

$$\begin{aligned} \dot{x}(t) &= f(x(t), u(t), v(t)), \quad x(0) = x_0,\ t \geq 0 \\ e(t) &= h(x(t), u(t), v(t)) \end{aligned} \tag{4.1}$$

where $x(t)$ is the plant state, $u(t)$ is the plant input, $e(t)$ is the plant output representing tracking error, and $v(t)$ are reference inputs and disturbances generated in an exosystem.

$$\dot{v}(t) = a(v) = Sv(t), \quad v(0) = v_0,\ t \geq 0 \tag{4.2}$$

We assume $f \in C^k(R^n \times R^m \times R^q)$, $h \in C^k(R^n \times R^m \times R^q)$, and $f(0,0,0) = 0$, $h(0,0,0) = 0$ where k is a sufficiently larger number.

The state feedback control law is of the form

$$u = \psi(x, v) \tag{4.3}$$

where $\psi \in C^k(R^n \times R^q)$ and $\psi(0,0) = 0$. The closed-loop system, by using the above control law (4.3), will be as follows

$$\begin{aligned} \dot{x}(t) &= f_c(x,v) \triangleq f(x,\psi(x,v),v) \\ e &= h_c(x,v) \triangleq h(x,\psi(x,v),v) \end{aligned} \tag{4.4}$$

The closed-loop system (4.4) can be described by Figure 4.1 [32].

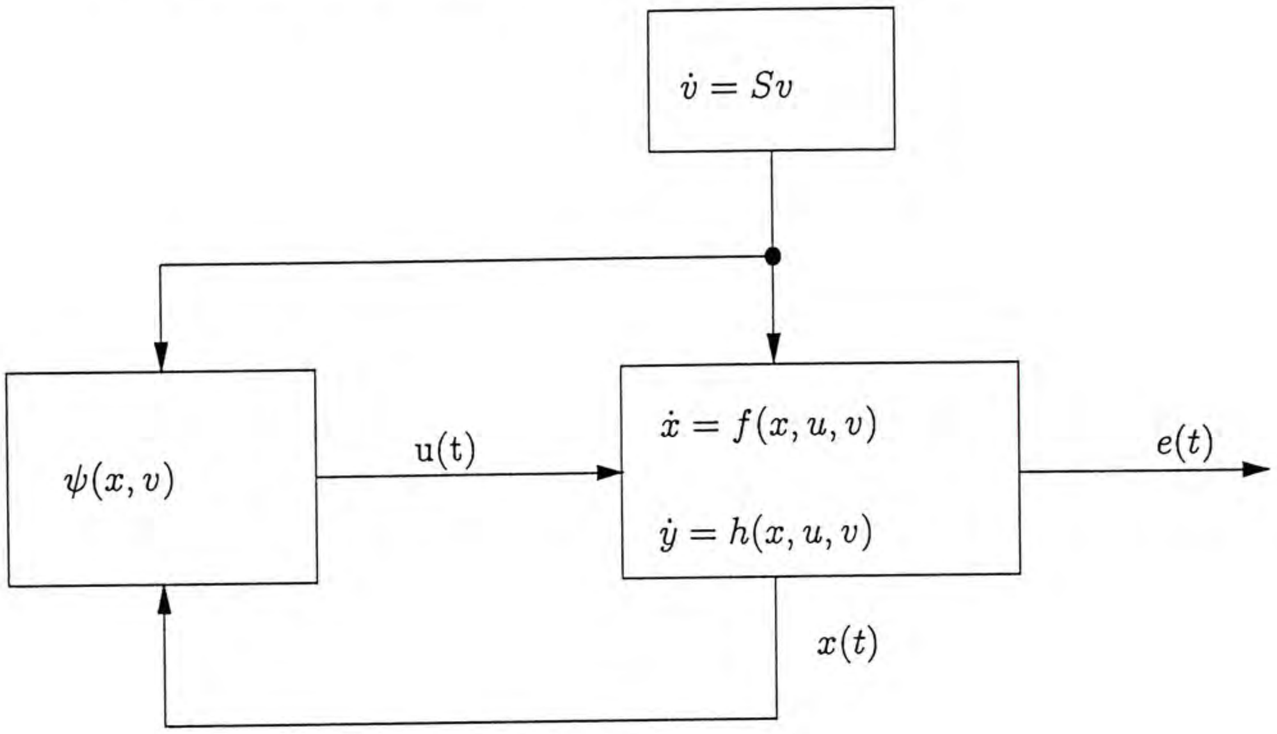


Figure 4.1: Block Diagram Representation of the Closed-Loop Nonlinear System

Output Regulation Problem : Design a feedback control law of the form $u = \psi(x,v)$ such that the closed-loop system satisfies

R1: The eigenvalues of $\frac{\partial f_c}{\partial x}(0,0)$ have negative real part.

R2: For sufficiently small x_0 and v_0, the solution of the closed-loop system exists for all $t \geq 0$, and

$$\lim_{t\to\infty} ||h_c(x(t),v(t))|| = 0 \tag{4.5}$$

The first requirement ensures the (local) asymptotic stability of the closed-loop system [32], which ensures the (local) bounded-input bounded state property for the closed-loop system (4.4).

The solvability of the above problem is given as follows:

Theorem 1: Assume

A1: The pair $\{\frac{\partial f}{\partial x}(0,0,0),\ \frac{\partial f}{\partial u}(0,0,0)\}$ is stabilizable, and

A2: The equilibrium of exosystem (4.2) at the origin is stable, and all the eigenvalues of S lies on the imaginary axis.

A3: There exist two sufficiently smooth functions $\mathbf{x}(v)$ and $\mathbf{u}(v)$ defined in an open neighborhood V of the origin of R^q such that $\mathbf{x}(0) = 0$, $\mathbf{u}(0) = 0$, and, for all $v \in V$,

$$\frac{\partial \mathbf{x}(v)}{\partial v} Sv = f(\mathbf{x}(v), \mathbf{u}(v), v) \tag{4.6}$$

$$0 = h(\mathbf{x}(v), \mathbf{u}(v), v) \tag{4.7}$$

The above two equations (4.6),(4.7) are called the regulator equations. The solution of these two equations leads to a state feedback control law given by [32]

$$u(t) = \mathbf{u}(v(t)) + K[x(t) - \mathbf{x}(v(t))] \tag{4.8}$$

where K is a feedback gain such that all the eigenvalues of the matrix

$$\frac{\partial f}{\partial x}(0,0,0) + \frac{\partial f}{\partial u}(0,0,0)K \tag{4.9}$$

have negative real parts which always exists under assumption $A1$.

4.2 Reduction of Regulator Equation into Center Manifold Equations

Equations (4.6) and (4.7) are known as the regulator equations. For a large class of nonlinear systems, the solvability of regulator equations (4.6) and (4.7) can be reduced to the solvability of a center manifold equation of the form (3.1). Consider a single-input-single-output system described in the standard form

$$\begin{aligned} \dot{x}(t) &= f(x) + g(x)u,\ x \in R^n,\ u \in R,\ y \in R \\ y &= h(x) \end{aligned} \tag{4.10}$$

Assuming the relative degree of (4.10) is r, then its normal form is given by

$$\begin{aligned} \dot{x}_1 &= x_2 \\ &\vdots \end{aligned}$$

$$\begin{aligned}
\dot{x}_{r-1} &= x_r \\
\dot{x}_r &= \sigma(x) + \rho(x)u \\
\dot{x}_{r+1} &= q_1(x) \\
&\vdots \\
\dot{x}_n &= q_{n-r}(x) \\
y &= x_1 \qquad (4.11)
\end{aligned}$$

where $\sigma(x)$, $\rho(x)$, $q_1(x), \cdots, q_{n-r}(x)$ are some sufficiently smooth functions.

Consider the tracking problem for a reference input $d(v)$ where $v \in R^q$ is generated by

$$\dot{v} = Sv \qquad (4.12)$$

with $S \in R^{q\times q}$ a matrix with all its eigenvalues on the imaginary axis. Thus, the error equation is defined as

$$e = y - d(v) \qquad (4.13)$$

The regulator equations corresponding to (4.11), (4.12) and (4.13) are

$$\begin{aligned}
\frac{\partial \mathbf{x}_1(v)}{\partial v}Sv &= \mathbf{x}_2(v) && (4.14) \\
&\vdots \\
\frac{\partial \mathbf{x}_{r-1}(v)}{\partial v}Sv &= \mathbf{x}_r(v) && (4.15) \\
\frac{\partial \mathbf{x}_r(v)}{\partial v}Sv &= \sigma(\mathbf{x}(v)) + \rho(\mathbf{x}(v))\mathbf{u}(v) && (4.16) \\
\frac{\partial \mathbf{x}_{r+1}(v)}{\partial v}Sv &= q_1(\mathbf{x}_1(v), \cdots, \mathbf{x}_r(v), \mathbf{x}_{r+1}(v), \cdots, \mathbf{x}_n(v)) && (4.17) \\
&\vdots \\
\frac{\partial \mathbf{x}_n(v)}{\partial v}Sv &= q_{n-r}(\mathbf{x}_1(v), \cdots, \mathbf{x}_r(v), \mathbf{x}_{r+1}(v), \cdots, \mathbf{x}_n(v)) && (4.18) \\
0 &= \mathbf{x}_1(v) - d(v) && (4.19)
\end{aligned}$$

Equations (4.19), and (4.13) to (4.16) give

$$\begin{aligned}
\mathbf{x}_1(v) &= d(v) && (4.20) \\
\mathbf{x}_2(v) &= \frac{\partial \mathbf{x}_1(v)}{\partial v}Sv && (4.21) \\
\mathbf{x}_3(v) &= \frac{\partial \mathbf{x}_2(v)}{\partial v}Sv && (4.22)
\end{aligned}$$

$$\vdots$$

$$\mathbf{x}_r(v) = \frac{\partial \mathbf{x}_r(v)}{\partial v} S v \tag{4.23}$$

and

$$\mathbf{u}(v) = \frac{\frac{\partial \mathbf{x}_r(v)}{\partial v} S v - \sigma(\mathbf{x}(v))}{\rho(\mathbf{x}(v))} \tag{4.24}$$

Let

$$z = \begin{bmatrix} x_{r+1} \\ \vdots \\ x_n \end{bmatrix}, \quad \alpha(z, v) = \begin{bmatrix} q_1(\mathbf{x}_1(v), \cdots, \mathbf{x}_r(v), z) \\ \vdots \\ q_{n-r}(\mathbf{x}_1(v), \cdots, \mathbf{x}_r(v), z) \end{bmatrix} \tag{4.25}$$

Then equations (4.17) to (4.18) can be put in the following standard form

$$\frac{\partial \mathbf{z}(v)}{\partial v} S v = \alpha(\mathbf{z}(v), v) \tag{4.26}$$

where $\mathbf{z}(v) = \begin{bmatrix} \mathbf{x}_{r+1}(v) \\ \vdots \\ \mathbf{x}_n(v) \end{bmatrix}$. Clearly (4.26) is a center manifold equations of the system

$$\dot{z} = \alpha(z, v) \tag{4.27}$$

$$\dot{v} = S v \tag{4.28}$$

The solvability of (4.26) implies the solvability of (4.14) to (4.19) and leads to the solvability of the output regulation problem. Thus, equations (4.6) and (4.7) can be reduced to center manifold equations of the form (3.1). In other words, the solution of the regulator equations (4.6) and (4.7) are available as long as we can obtain the solution of the center manifold equations. From Chapter 3, we have shown that the approximated solution of center manifold equations can be obtained by neural based method. Therefore, with using the approximated solution of the center manifold equations, the approximated solution of the regulator equations is obtained which results in the solvability of the output regulation problem. Therefore, in Chapter 5, we will illustrate the approximate tracking of a reference input in the ball and beam. In addition, in Chapter 6, we will demonstrate the neural approximate disturbance rejection of the TORA system which involves the approximated solution of the center manifold equations.

Chapter 5

Application to the Control Design of Ball and Beam System

5.1 Problem Description

The ball and beam system is shown in Figure 5.1. The system dynamics can be described

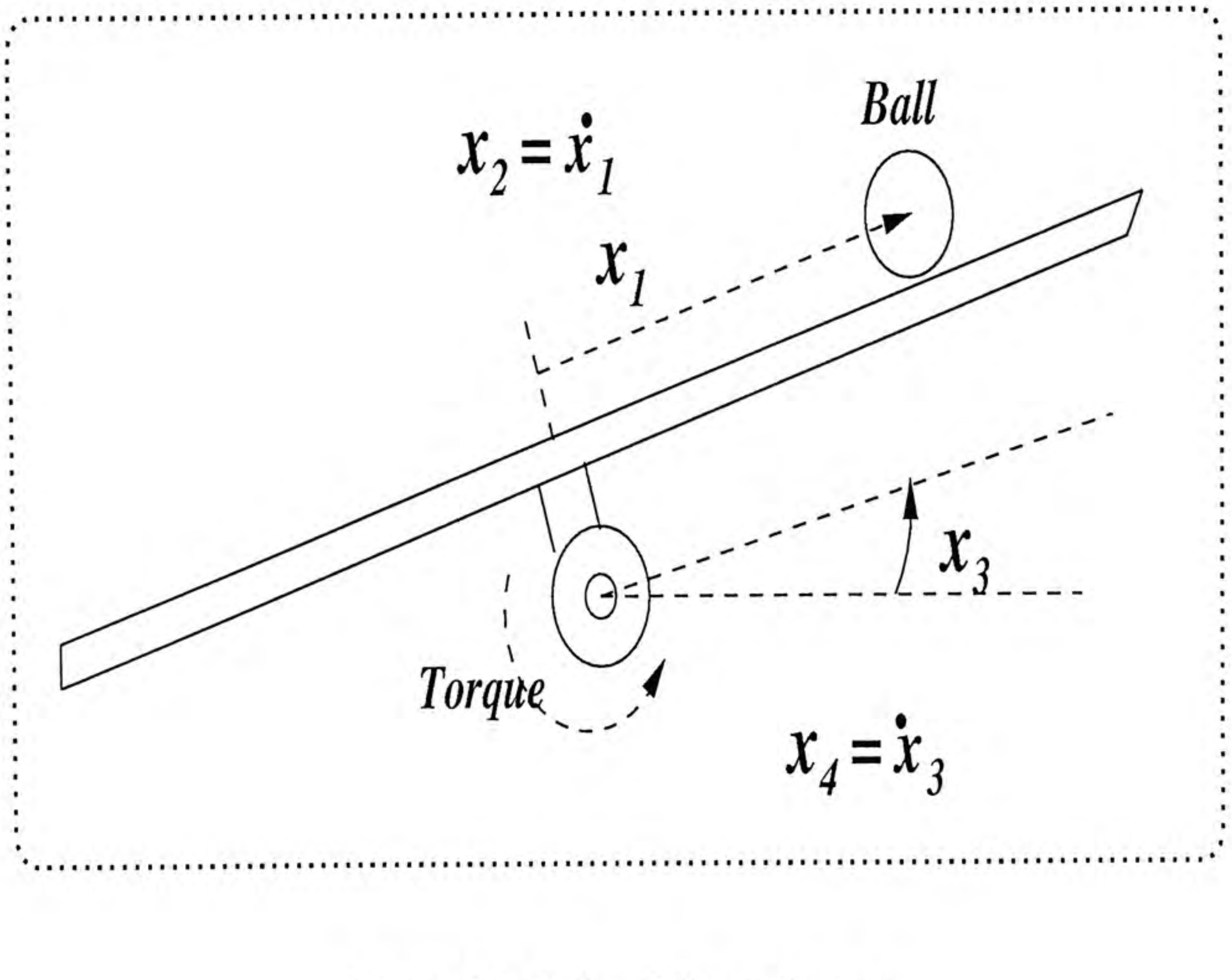


Figure 5.1: Ball and Beam System

by

$$\begin{aligned}
\dot{x}_1 &= x_2(t) \\
\dot{x}_2 &= Bx_1(t)x_4^2(t) - BGsinx_3(t) \\
\dot{x}_3 &= x_4(t) \\
\dot{x}_4 &= u \\
y(t) &= x_1(t) \qquad (5.1)
\end{aligned}$$

where y is the ball position, $G = 9.81m/s^2$ is the acceleration of gravity, and $B = 0.7143$ some dimensionless constant [17].

We will consider the problem of designing a state feedback control law for this system such that the position of the ball can asymptotically track a sinusoidal function $y_d(t) = Asin\omega t$. Let us put (5.1) into the following standard form

$$\begin{aligned}
\dot{x} &= f(x) + g(x)u \\
y &= h(x) \qquad (5.2)
\end{aligned}$$

where

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}, \quad f(x) = \begin{bmatrix} x_2 \\ Bx_1x_4^2 - BGsin(x_3) \\ x_4 \\ 0 \end{bmatrix}, \quad g(x) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

and $h(x) = x_1$.

Also, introduce the following exosystem

$$\dot{v} = Sv, \ \ t \geq 0, \ \ v(0) = v_0 \qquad (5.3)$$

with

$$v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}, \quad S = \begin{bmatrix} 0 & \omega \\ -\omega & 0 \end{bmatrix}, \quad v_0 = \begin{bmatrix} 0 \\ A \end{bmatrix}$$

Then clearly, $y_d(t) = v_1$.

The regulator equations associated with the above tracking problem take the following form:

$$\frac{\partial \mathbf{x}(v)}{\partial v} Sv = f(\mathbf{x}(v)) + g(\mathbf{x}(v))\mathbf{u}(v) \qquad (5.4)$$

$$0 = \mathbf{x}_1(v) - v_1 \qquad (5.5)$$

Expanding (5.4) and (5.5) gives

$$
\begin{aligned}
\frac{\partial \mathbf{x}_1(v)}{\partial v}Sv &= \mathbf{x}_2(v) &\quad (5.6)\\
\frac{\partial \mathbf{x}_2(v)}{\partial v}Sv &= B\mathbf{x}_1(v)\mathbf{x}_4(v)^2 - BGsin\mathbf{x}_3(v) &\quad (5.7)\\
\frac{\partial \mathbf{x}_3(v)}{\partial v}Sv &= \mathbf{x}_4(v) &\quad (5.8)\\
\frac{\partial \mathbf{x}_4(v)}{\partial v}Sv &= \mathbf{u}(v) &\quad (5.9)\\
\mathbf{x}_1(v) &= v_1 &\quad (5.10)
\end{aligned}
$$

from which we can obtain

$$
\begin{aligned}
\mathbf{x}_1(v) &= v_1 &\quad (5.11)\\
\mathbf{x}_2(v) &= \omega v_2 &\quad (5.12)\\
\mathbf{u}(v) &= \frac{\partial \mathbf{x}_4(v)}{\partial v}Sv &\quad (5.13)
\end{aligned}
$$

with two unknown functions $\mathbf{x}_3(v)$ and $\mathbf{x}_4(v)$ satisfying

$$
\begin{aligned}
-\omega^2 v_1 &= Bv_1\mathbf{x}_4(v)^2 - BGsin\mathbf{x}_3(v) &\quad (5.14)\\
\frac{\partial \mathbf{x}_3(v)}{\partial v}Sv &= \mathbf{x}_4(v) &\quad (5.15)
\end{aligned}
$$

It is clear that in order to obtain the solution of regulator equations (5.4) and (5.5), we only need to obtain the solution of the equation (5.15) where $\mathbf{x}_4(v)$ satisfies $-\omega v_1 = B\mathbf{x}_1(v)\mathbf{x}_4(v)^2 - BGsin\mathbf{x}_3(v)$. Thus

$$
\begin{aligned}
\mathbf{x}_4(v) &= \sqrt{\frac{-\omega v_1 + BGsin\mathbf{x}_3(v)}{Bv_1}}\\
&= \alpha(\mathbf{x}_3(v), v)
\end{aligned}
$$

Let

$$\mathbf{z}(v) = \mathbf{x}_3(v) \text{ and } \alpha(\mathbf{z}(v), v) = \sqrt{\frac{-\omega v_1 + BGsin\mathbf{x}_3(v)}{Bv_1}} \quad (5.16)$$

which is in the form of (3.1). Once we can obtain the solution of (5.14) and (5.15), we can obtain the solution of the regulator equations by (5.11) to (5.13). Next, we will use neural based approximation method described in last section to solve (5.14) and (5.15).

5.2 Neural Approximation Solution of Center Manifold Equations

By using the same network model which is shown in the previous section, we approximate the unknown functions $\mathbf{x}_3(v)$ and $\mathbf{x}_4(v)$ as the outputs of the neural model. That is,

$$\mathbf{x}_3(W,v) = \sum_{i=1}^{N} w_i^{O1}\phi(y_i) \tag{5.17}$$

$$\mathbf{x}_4(W,v) = \sum_{i=1}^{N} w_i^{O2}\phi(y_i) \tag{5.18}$$

where

$$y_i = \sum_{j=1}^{2} w_{ij}^{I}v_i + w_{i0}^{I} \tag{5.19}$$

and N is number of hidden neurons, and $\phi(.)$ is the sigmoid activation function. $\mathbf{x}_3(W,v)$ and $\mathbf{x}_4(W,v)$ are the approximated solutions of $\mathbf{x}_3(v)$ and $\mathbf{x}_4(v)$ respectively. An objective function is defined which is composed of two error functions e_1 and e_2 in terms of the approximated function $\mathbf{x}_3(W,v)$ and $\mathbf{x}_4(W,v)$. Those equations are defined as

$$e_1 = -\omega^2 v_1 - B\mathbf{x}_1(v)\mathbf{x}_4(W,v)^2 + BGsin\mathbf{x}_3(W,v) \tag{5.20}$$

$$e_2 = \frac{\partial \mathbf{x}_3(W,v)}{\partial v}Sv - \mathbf{x}_4(W,v) \tag{5.21}$$

$$J(W,v) = \frac{1}{2}(e_1^2 + e_2^2) \tag{5.22}$$

$$Q(W) = \sum_{v\in\Gamma_d} J(W,v) \tag{5.23}$$

The gradients of the above equations with respect to different scalars, or called weights, are

$$\frac{\partial e_{1p}}{\partial w_i^{O1}} = GB\phi(z_i)cos\mathbf{x}_3(W,v) \tag{5.24}$$

$$\frac{\partial e_{2p}}{\partial w_i^{O1}} = \sum_{i=1}^{N}\phi'(z_i)w_{i1}^{I}\omega v_2 - \phi'(z_i)w_{i2}^{I}\omega v_1 \tag{5.25}$$

$$\frac{\partial e_{1p}}{\partial w_i^{O2}} = -2Bv_1\mathbf{x}_4(W,v)\phi(z_i) \tag{5.26}$$

$$\frac{\partial e_{2p}}{\partial w_i^{O2}} = -\phi(z_i) \tag{5.27}$$

$$\frac{\partial e_{1p}}{\partial w_{ij}^{I}} = -2Bv_1\mathbf{x}_4(W,v)(\phi'(z_i)w_i^{O2}v_j) + GBcos\mathbf{x}_3(W,v)(\phi'(z_i)w_i^{O1}v_j) \tag{5.28}$$

$$\frac{\partial e_{2p}}{\partial w_{i1}^{I}} = [-2w_i^{O1}\phi(z_i)\phi'(z_i)v_1w_{i1}^{I} + w_i^{O1}\phi'(z_i)]\omega v_2$$

$$+[-2w_i^{O1}\phi(z_i)\phi'(z_i)v_1w_{i2}^I]\omega v_1 - w^O2_iv_1\phi'(z_i) \tag{5.29}$$

$$\begin{aligned}\frac{\partial e_{2p}}{\partial w_{i2}^I} &= [-2w_i^{O1}\phi(z_i)\phi'(z_i)v_2w_{i1}^I + w_i^{O1}\phi'(z_i)]\omega v_2 \\ &\quad +[-2w_i^{O1}\phi(z_i)\phi'(z_i)v_2w_{i2}^I]\omega v_1 - w^O2_iv_2\phi'(z_i)\end{aligned} \tag{5.30}$$

Thus, the desirable weight W can be searched by using the gradient descent method as described in chapter 2. We have tested the case for $\omega = \pi/2$, $N = 20$, $\Gamma = \{v \in R^2 \mid ||v|| \leq 2\}$, and $\Gamma_d = \{(\rho sin\theta, \rho cos\theta) \mid \rho = 0.1, 0.2, \cdots 2, \theta = -\pi, -0.9\pi, -0.8\pi \cdots, 0, 0.1\pi \cdots, \pi\}$. Figure 5.2 shows the performance function $Q(W)$ versus the iteration steps. The algorithm stops when $Q(W) < \epsilon_r = 10^{-2}$.

5.3 Simulation Results

Now we are going to show the system performance from different approximation approaches.

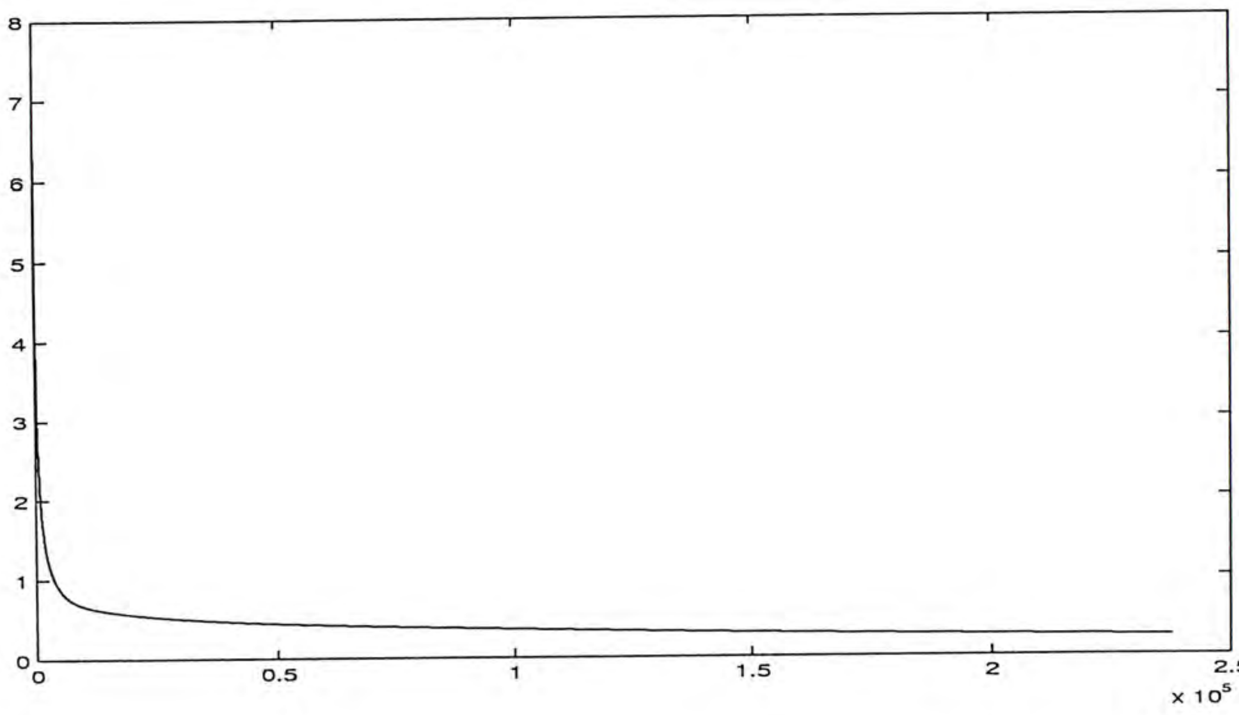


Figure 5.2: The Performance Function $Q(W)$ Against Training Iteration

Our proposed method has been already mentioned clearly, so it is time for describing the other two approximation approaches. From [39], there is an brief description of the 1^{st}

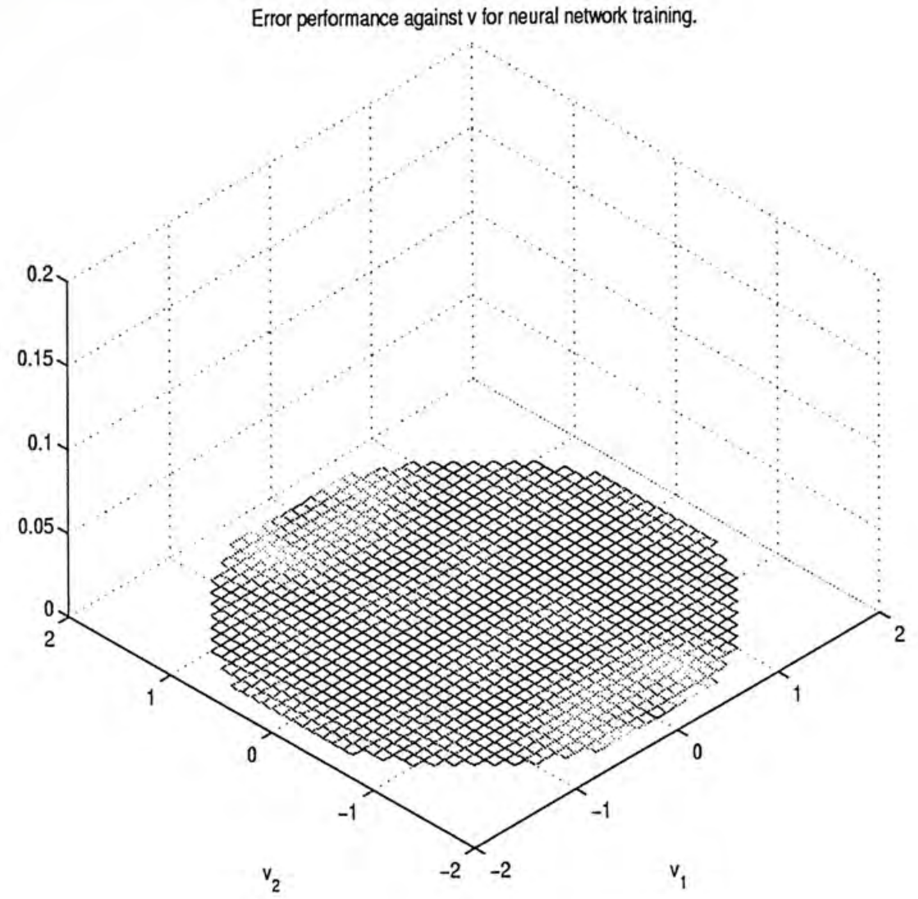


Figure 5.3: The Performance Surfaces of Neural Based Approximation $J(W, v)$

order and 3^{rd} order Taylor series approximation. The Taylor series of $\hat{\mathbf{x}}(v)$ is

$$\hat{\mathbf{x}}(v) = \begin{bmatrix} v_1 \\ \omega v_2 \\ a_1 v_1 + a_{12} v_1 v_2^2 + a_{30} v_1^3 + \cdots \\ b_1 v_2 + b_{21} v_1^2 v_2 + b_{03} v_2^3 + \cdots \end{bmatrix} \tag{5.31}$$

where

$$\begin{array}{lll} a_1 = \frac{\omega^2}{BG} & a_{12} = \frac{\omega^6}{B^2G^3} & a_{30} = \frac{\omega^6}{6B^3G^3} \\ b_1 = \frac{\omega^3}{BG} & b_{21} = \frac{\omega^7 - 4B\omega^7}{2B^3G^3} & b_{03} = \frac{\omega^7}{B^2G^3} \end{array}$$

The performance surfaces from the Taylor based approach are show in Figures 5.4 and 5.5.

There is a sharp increase in performance value at outer region among three surfaces. However, the peak value of our neural based approximated result is smaller than those from other two approximation approaches. On the other hand, the performance surface based on our proposed method is more smooth than other two approaches. It reflects that the neural based approach can get a smaller performance value throughout all the region

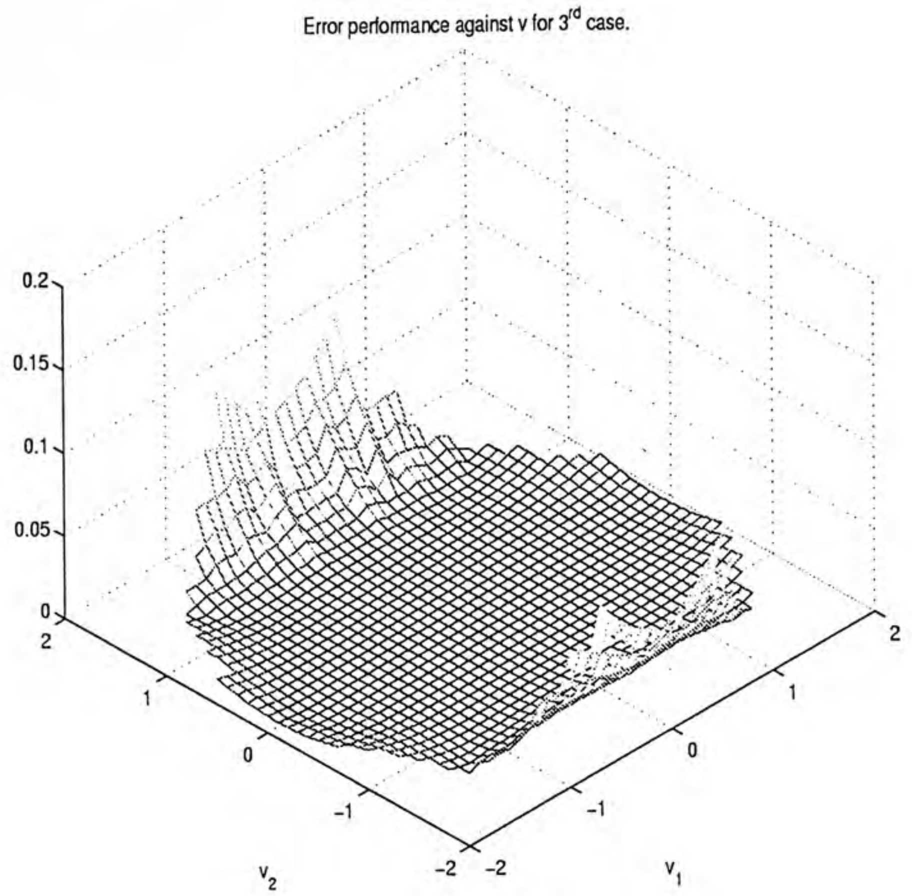


Figure 5.4: The Performance Surface of 3^{rd} order Approximation $J^3(v)$

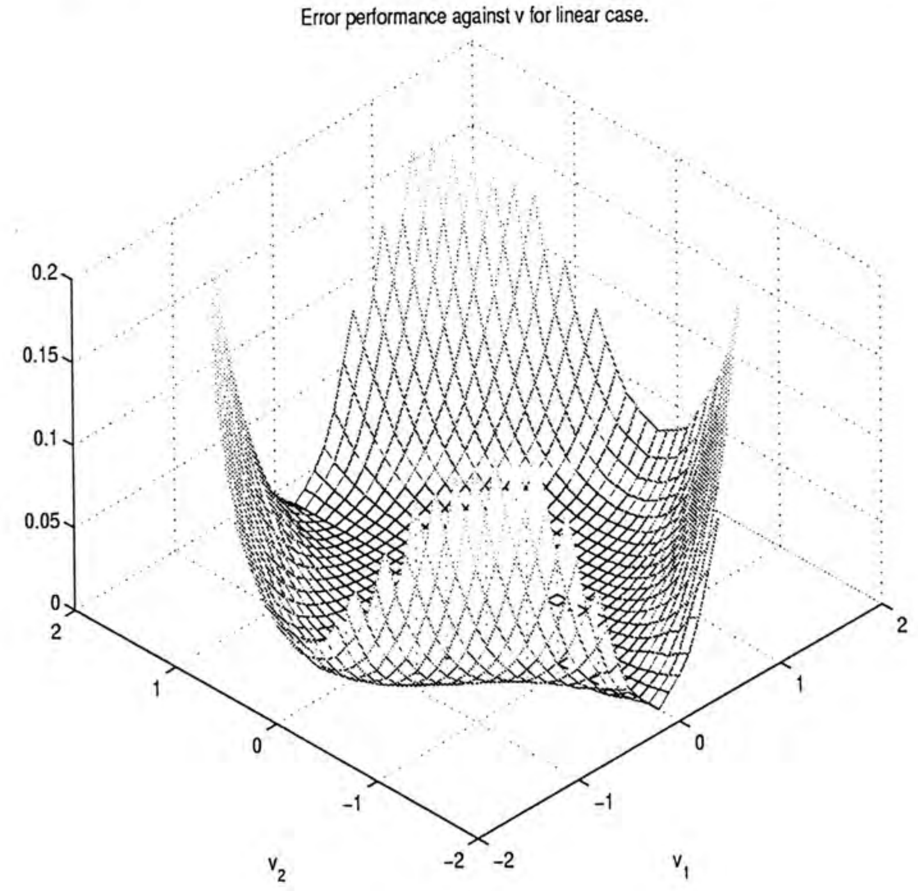


Figure 5.5: The Performance Surface of 1^{st} order Approximation $J^1(v)$

while those from Taylor approach can only keep smaller value closer to the origin. Next, we draw our attention in the simulation result of the asymptotic tracking of the ball and beam system.

The overall control law consists of the approximated solution of $\mathbf{x}(v)$ and the feedback gain K. It is derived from the linearized system of ball and beam system (5.1). Refer to the linearized Jacobian matrix

$$\frac{\partial f(0)}{\partial x} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -BG & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \; g(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

The control law of 1st order and 3rd order approximation, drived from equation (5.31) with respect to the degree of approximation i, is thus defined. Therefore,

$$u = \mathbf{u}^i(v) + K[x - \mathbf{x}^i(v)], \; i = 1, 3$$

where $\mathbf{u}^1(v) = c_{10}v_1$ and $\mathbf{u}^3(v) = c_{10}v_1 + c_{12}v_1v_2^2 + c_{30}v_1^3$ with

$$c_{10} = -\tfrac{\omega^4}{BG}$$
$$c_{12} = \tfrac{(1-7B)\omega^8}{B^3G^3}$$
$$c_{30} = -\tfrac{\omega^8 - 4B\omega^8}{2B^3G^3}$$

The neural based control law is in the form of

$$u = \mathbf{u}(W, v) + K(x - \mathbf{x}(W, v)) \tag{5.32}$$

where $\mathbf{x}(W, v) = (\mathbf{x}_1(v), \mathbf{x}_2(v), \hat{\mathbf{x}}_3(W, v), \hat{\mathbf{x}}_4(W, v))$, and $\mathbf{u}(W, v) = \frac{\partial \hat{\mathbf{x}}_4(W,v)}{\partial v} Sv$. The feedback gain K is designed based on the Jacobian linearization of the ball and beam system. Using pole placement method gives $K = [-5.5812 \;\; -6.028 \;\; 21.25 \;\; 5.25]$ and $K = [-5.5973 \;\; -7.1597 \;\; 27.468 \;\; 7.81]$ which are such that the eigenvalues of $\frac{\partial f(0)}{\partial x} + g(0)K$ equal to $(-0.424 \pm 1.263i)\omega_0$, $(-0.626 \pm 0.4141i)\omega_0$ and $(-0.6573 \pm 0.8302i)\omega_0$, $(-0.9047 \pm 0.2711i)\omega_0$ where $\omega_0 = 2.5 rads^{-1}$. With the initial condition $x = [0 \; 0 \; 0 \; 0\,]^T$, the simulation results are shown as follows: Figures 5.6 to 5.8 show the system responses of the asymptotic tracking of a sinusoidal input with amplitude at 1.2. It is difficult to compare the performance from these three approaches by these figures. We thus focus on the steady state tracking error.

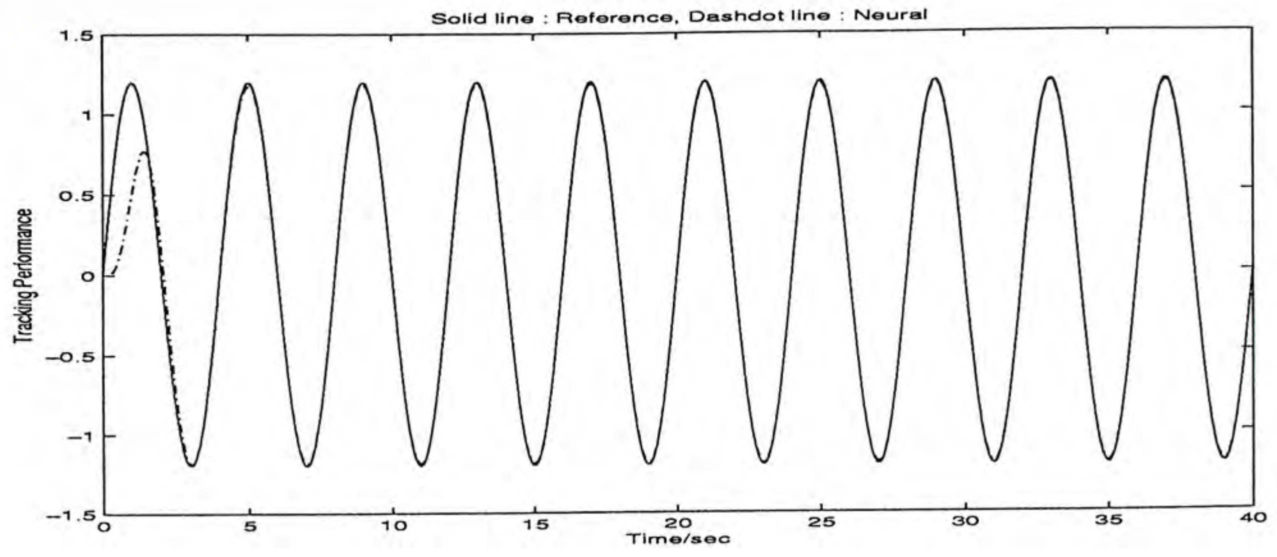


Figure 5.6: Time Response for Neural Based Tracking of a Sinusoidal Input at A=1.2

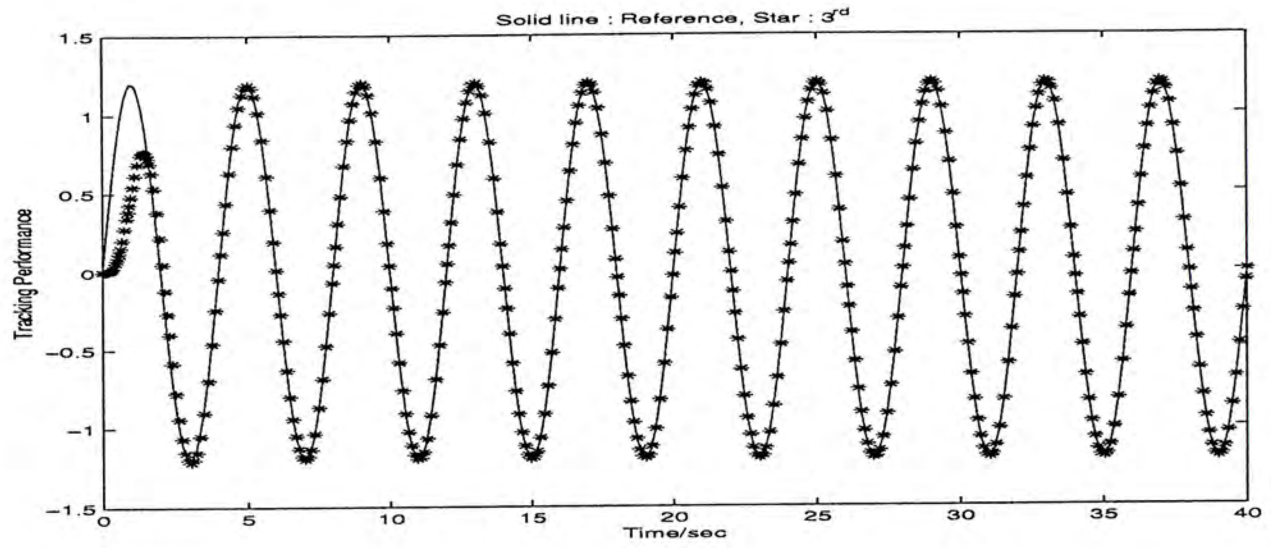


Figure 5.7: Time Response for 3^{rd} order Tracking of a Sinusoidal Input at A=1.2

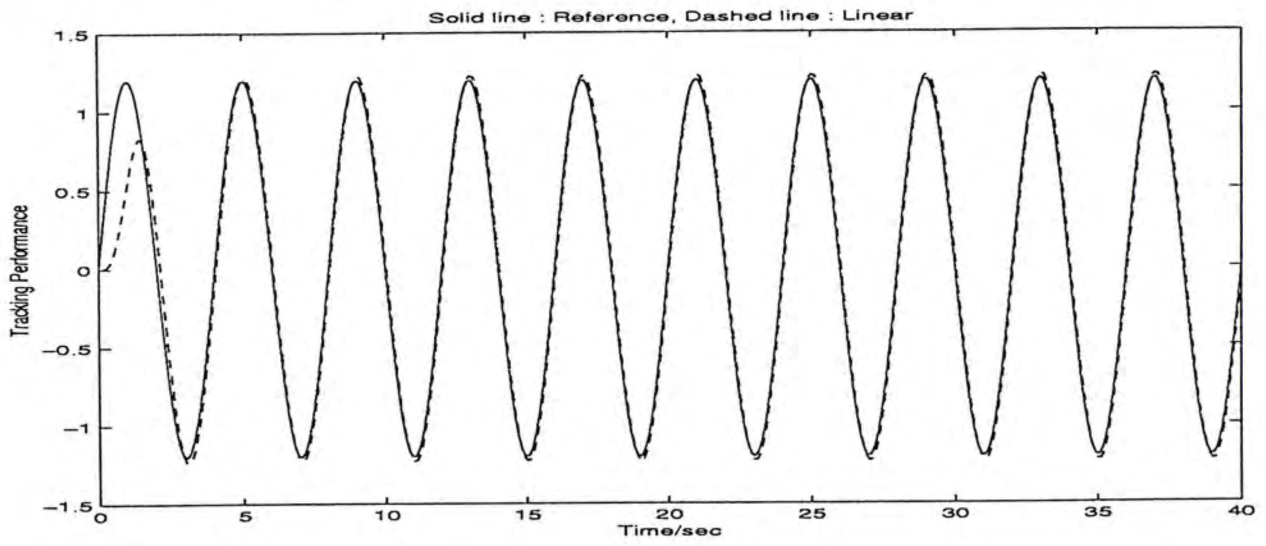


Figure 5.8: Time Response for Linearized Tracking of a Sinusoidal Input at A=1.2

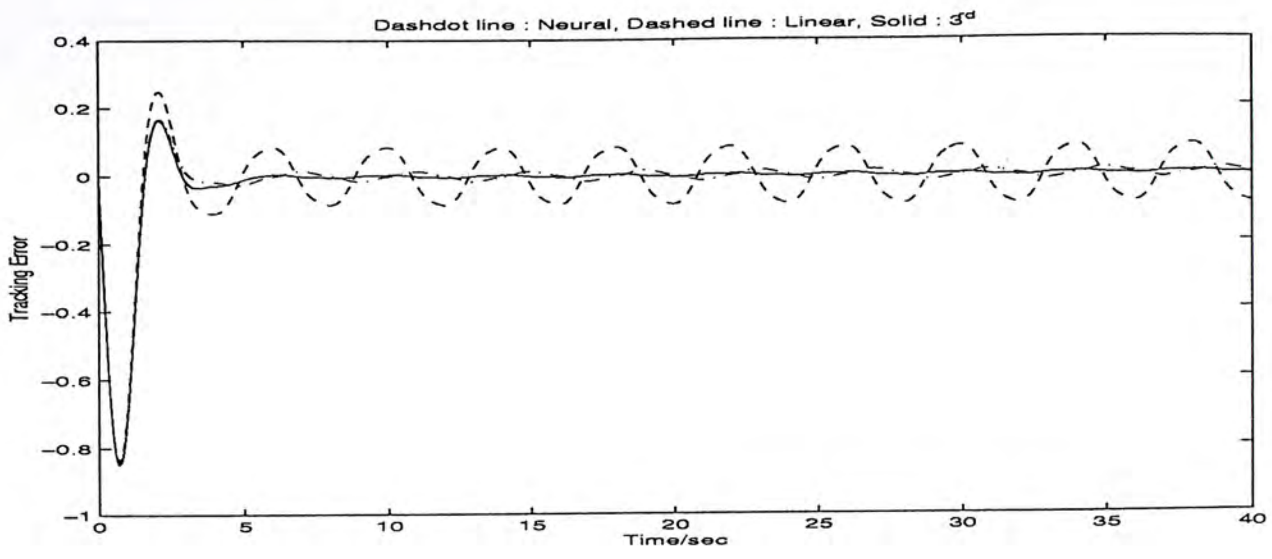


Figure 5.9: Tracking error at A=1.2

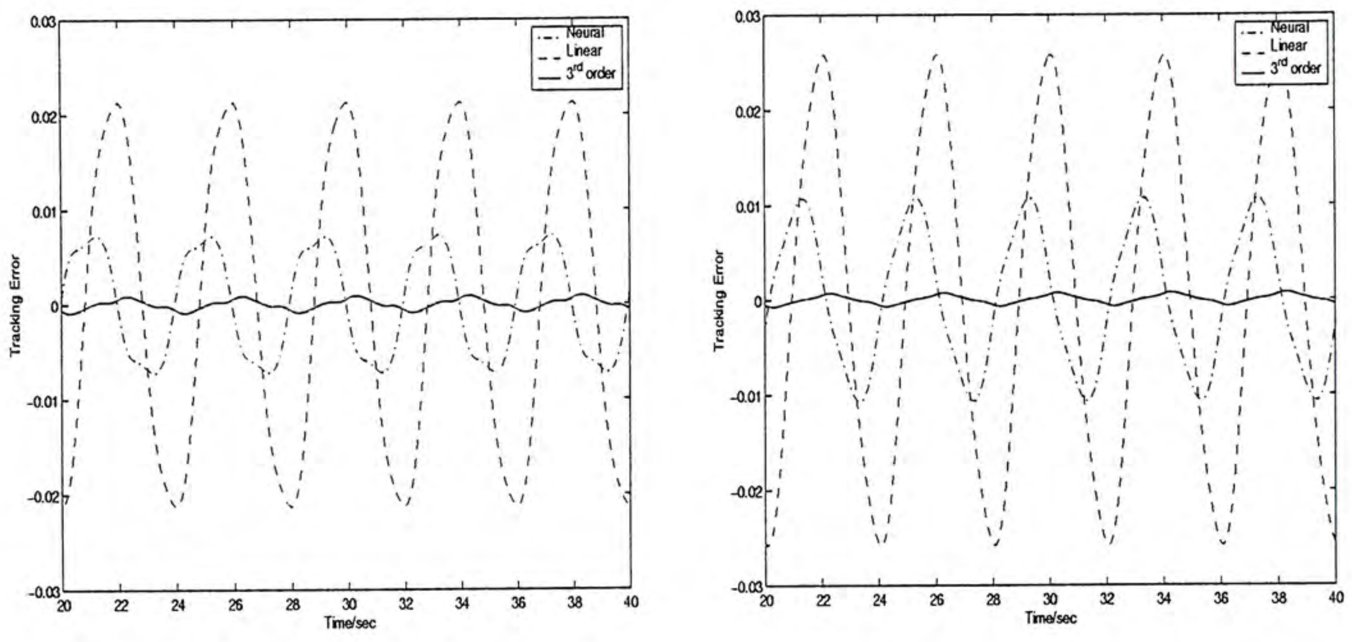


Figure 5.10: Steady State Tracking Error for A=0.8

Amplitude	Neural ITAE	Linear ITAE	3^{rd} ITAE	Neural BESSEL	Linear BESSEL	3^{rd} BESSEL
A=1.0	0.4%	4.66%	0.21%	0.43%	5.53%	0.2%
A=1.2	1.35%	7.42%	0.42%	1.59%	8.45%	0.42%
A=1.6	1.88%	17.16%	1.59%	2.18%	17.4%	1.61%
A=1.8	1.06%	26.4%	2.94%	1.18%	23.98%	2.87%
A=2.0	1.16%	40.98%	5.24%	1.24%	32.23%	4.81%

Table 5.1: Maximal Percentage Steady State Tracking Error

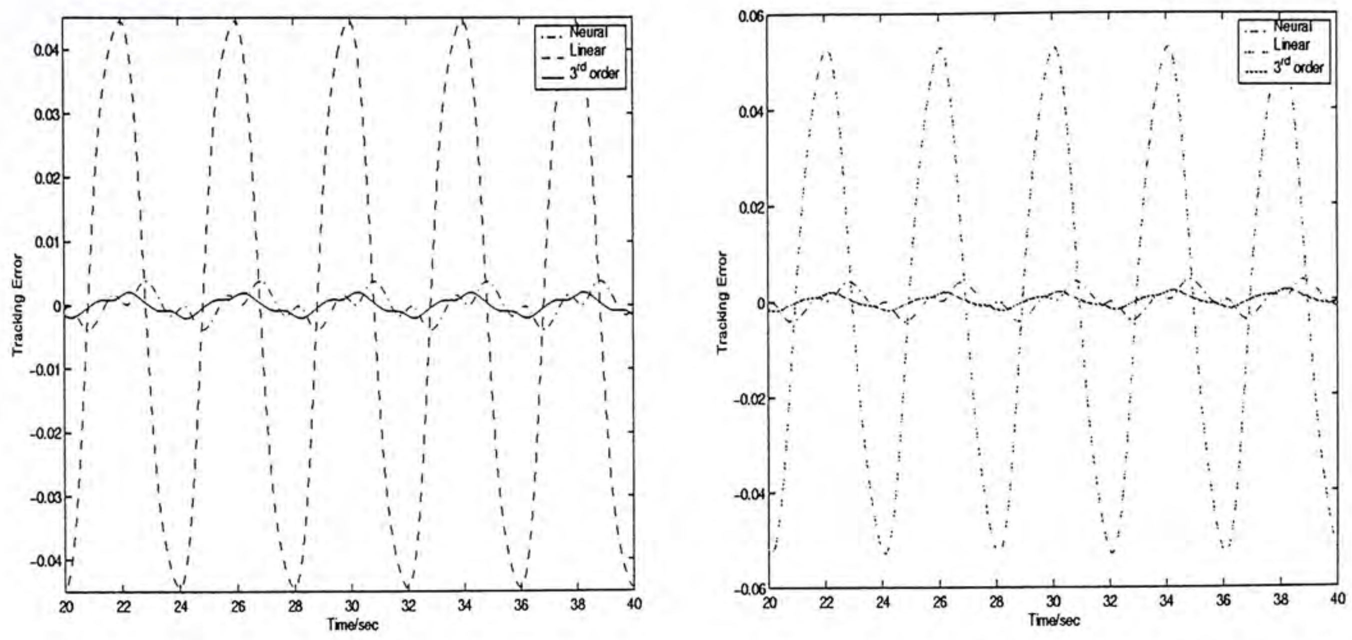


Figure 5.11: Steady State Tracking Error for A=1.0

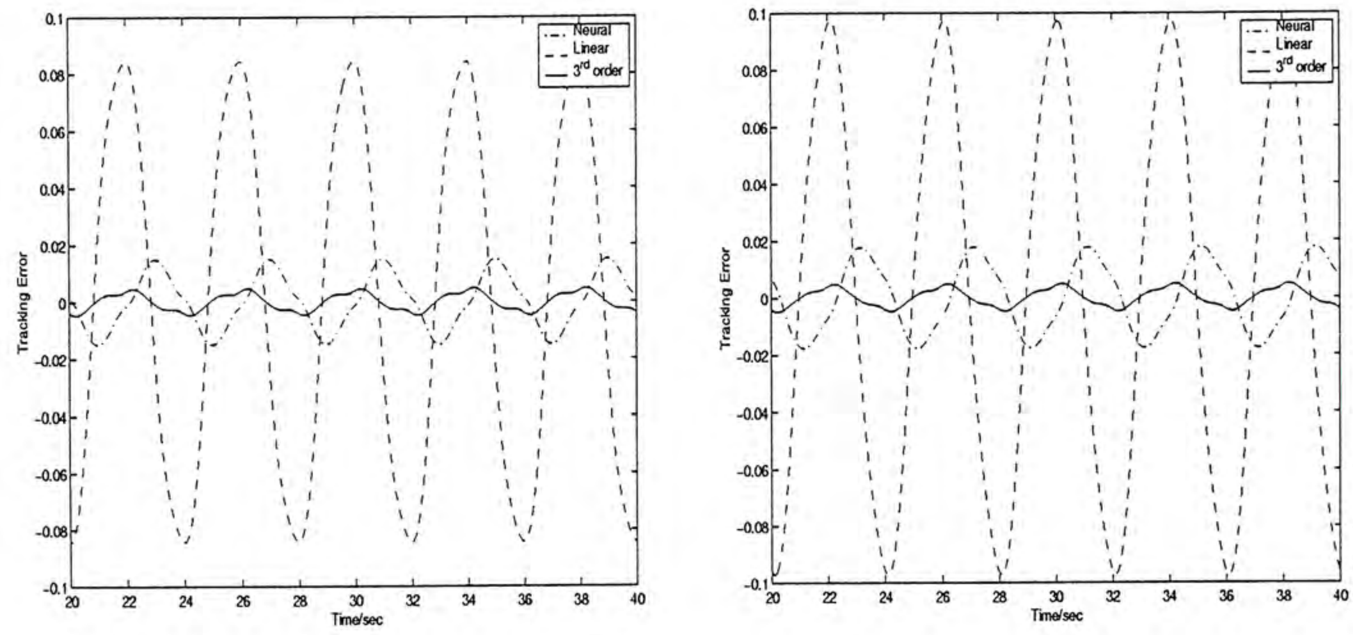


Figure 5.12: Steady State Tracking Error for A=1.2

Table 5.1 lists the maximal steady state tracking errors of the closed-loop system under the neural network based control, the linear control and the 3^{rd} order control with $\omega = \frac{\pi}{2}$ and $A = 1,\ 1.2,\ 1.6,\ 1.8,\ 2.0$. It is seen that the steady state tracking errors resulting from both Taylor approximation approaches increase significantly as the input amplitude A increase while the neural network based control law maintains a quite uniform performance over different amplitudes of the the sinusoidal signal. In the meantime, the tracking error from neural based approach is smaller than that from 3^{rd} order approach in case of the larger input amplitude, greater than $A = 1.6$. Figures 5.10 to 5.16 show all the tracking performance associated with three approaches.

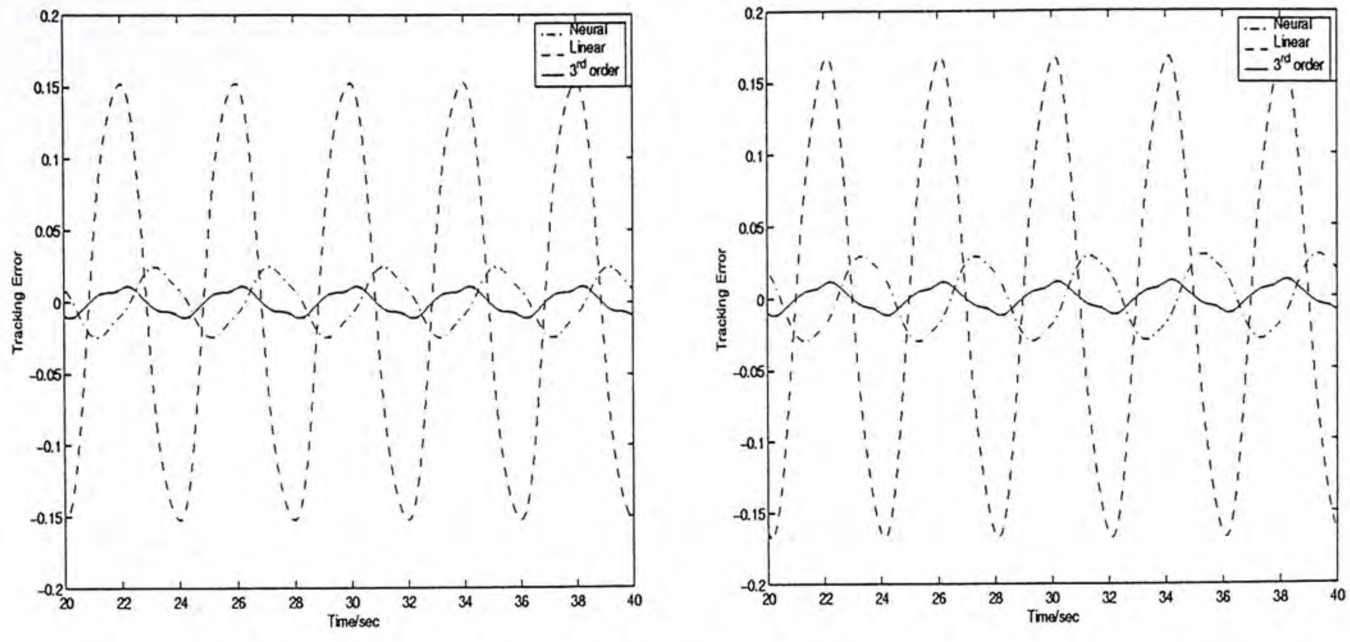


Figure 5.13: Steady State Tracking Error for A=1.4

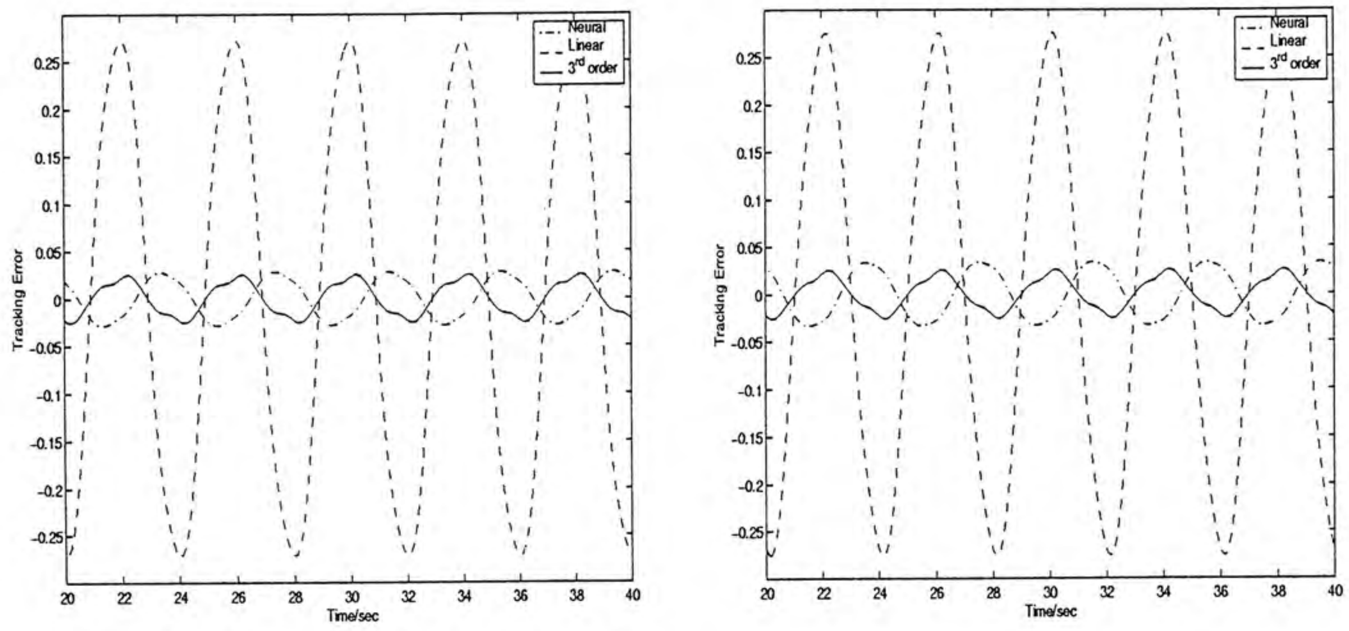


Figure 5.14: Steady State Tracking Error for A=1.6

5.4 Discussion

The choice of the feedback gain usually plays an important role for the system performance. In order to take the system performance in a systematic and logical way, we have chosen the feedback gain from two typical prototype design: ITAE and BESSEL [16].

ITAE prototype design is to minimize the integral of the time multiplied by the absolute value of the error:

$$\mathcal{I} = \int_0^\infty t|e|dt$$

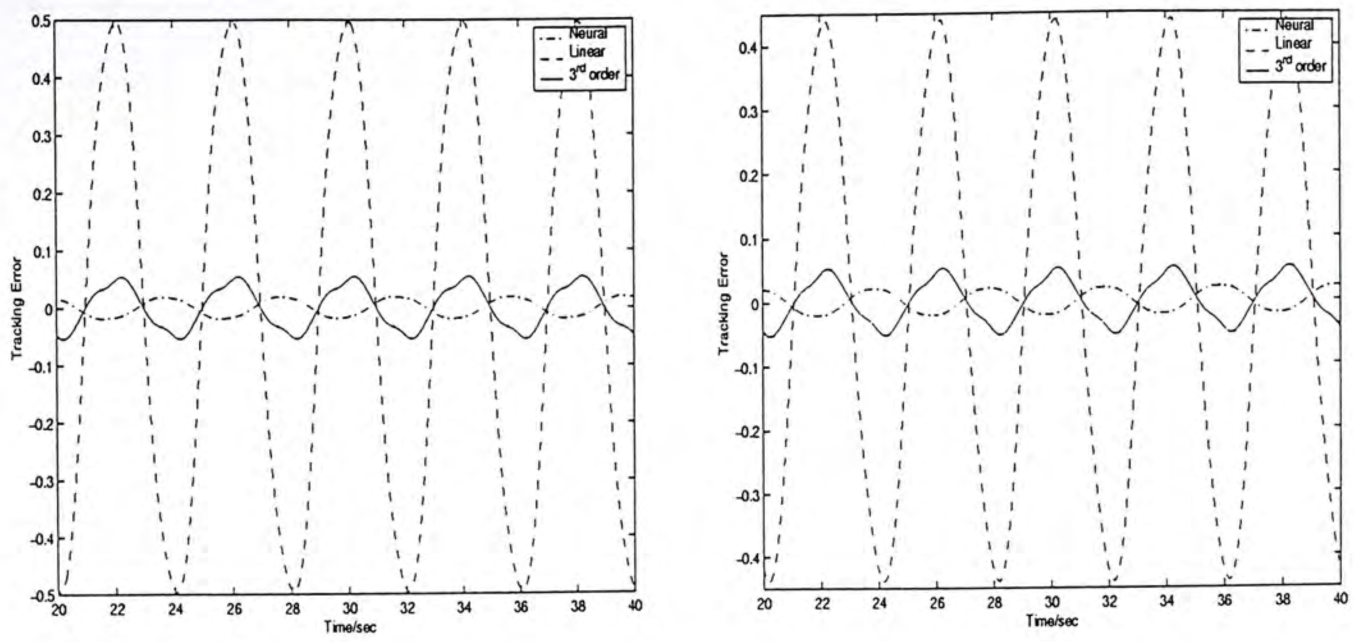


Figure 5.15: Steady State Tracking Error for A=1.8

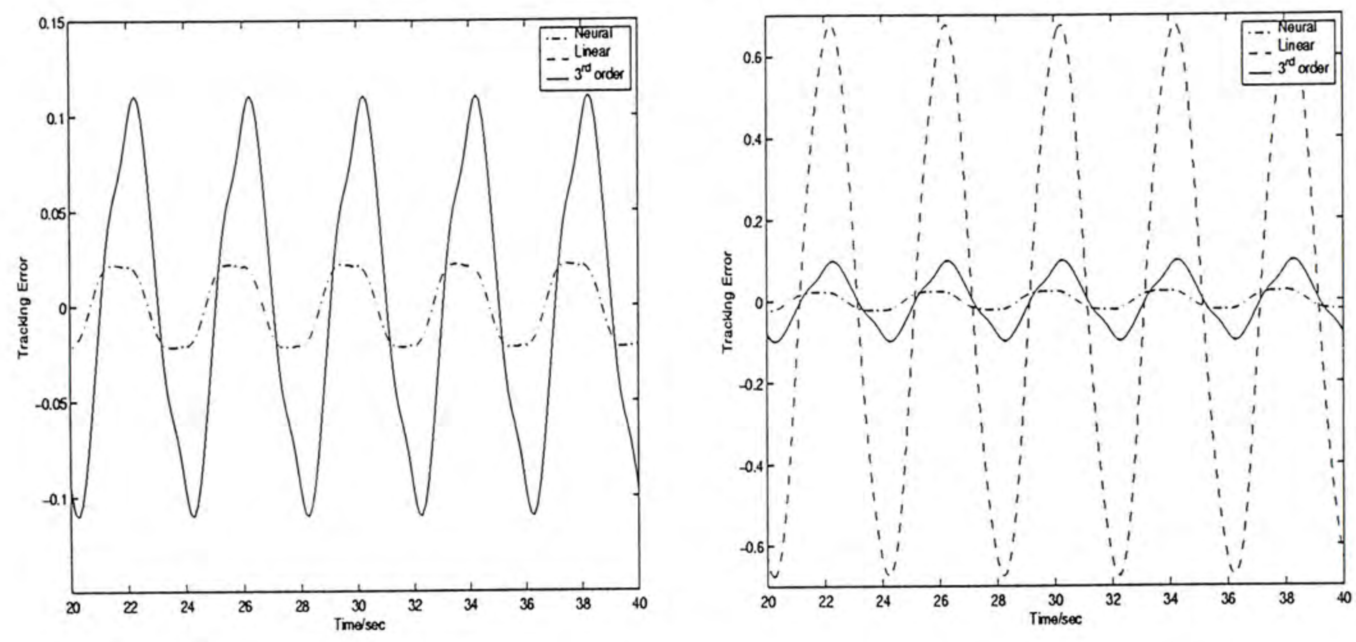


Figure 5.16: Steady State Tracking Error for A=2.0

Bessel filter prototype design is applied for those systems with all overshoots avoided. The transfer functions are given by $\frac{1}{B_n(s)}$, where $B_n(s)$ is the n^{th}-degree Bessel polynomial. Both prototype designs serve for those n^{th} orders systems.

Since the ball and beam system is a fourth order system, we choose the poles from Table 5.2 at $k = 4$. That is, $[(-0.424\omega_0 \pm 1.263\omega_0 i)\ (-0.626\omega_0 \pm 0.4141\omega_0 i)]$ and $[(-0.6573\omega_0 \pm 0.8302\omega_0 i)\ (-0.9047\omega_0 \pm 0.2711\omega_0 i)]$.

The design of the linearized feedback gain K affects the system response critically. For example, for the case $A = 0.2$, when the feedback gain is obtained from the Bessel prototyping, all three control approaches lead to stable closed-loop system, and when the

		k	Pole Locations for $\omega_0 = 1rads^{-1}$
(a)	ITAE transfer	1	$s+1$
	function poles	2	$s+0.7071 \pm 0.7071i$
		3	$(s+0.7081)(s+0.521 \pm 1.068i)$
		4	$(s+0.424 \pm 1.263i)(s+0.626 \pm 0.4141i)$
		5	$(s+0.8955)(s+0.3764 \pm 1.292i)(s+0.5758 \pm 0.5339i)$
		6	$(s+0.3099 \pm 1.2634i)(s+0.5805 \pm 0.7828i)(s+0.7346 \pm 0.2873i)$
(b)	Bessel transfer	1	$s+1$
	function poles	2	$s+0.866 \pm 0.5i$
		3	$(s+0.942)(s+0.7455 \pm 0.7112i)$
		4	$(s+0.6573 \pm 0.8302i)(s+0.9047 \pm 0.2711i)$
		5	$(s+0.9264)(s+0.5906 \pm 0.9072i)(s+0.8516 \pm 0.4427i)$
		6	$(s+0.5385 \pm 0.9617i)(s+0.7998 \pm 0.5622i)(s+0.9093 \pm 0.1856i)$

Table 5.2: Prototype Response Poles

feedback gain is obtained from the ITAE prototyping, the closed-loop system from the linear control law is unstable. Figure 5.16 shows the steady state tracking error where the left one is associated with the ITAE design while the right one Bessel design. Figure 5.17

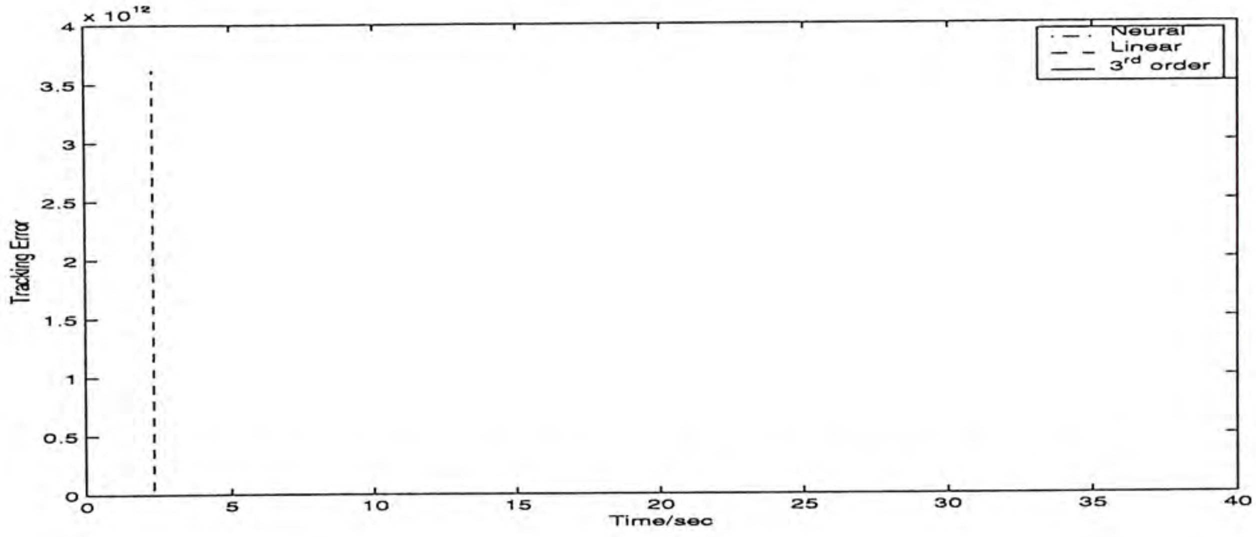


Figure 5.17: Time Response for Neural Based Tracking of a Sinusoidal Input at A=2.0

shows the time response of the closed-loop system with ITAE design.

Chapter 6

Neural Based Disturbance Rejection of Nonlinear Benchmark Problem (TORA System)

6.1 Problem Description

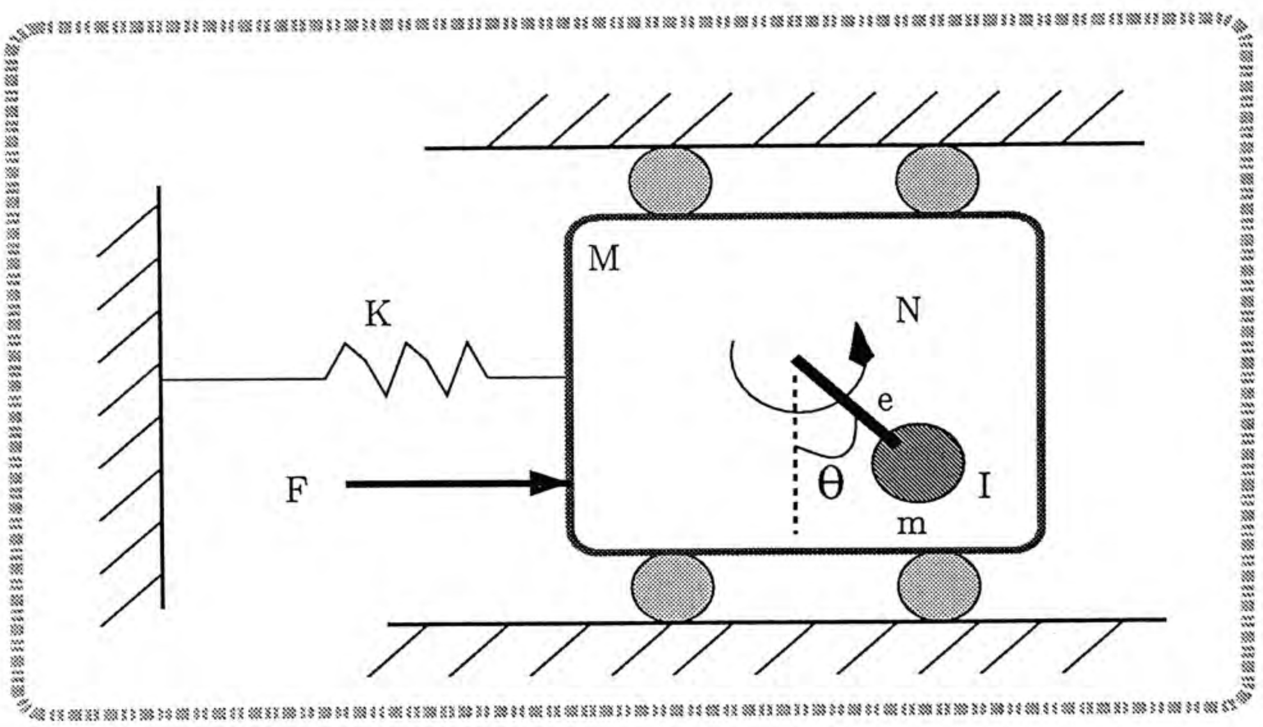


Figure 6.1: The TORA(Translational-Oscillational-Rotational Actuator) System

The TORA system has been described by P. Tsiotras, M. Corless and M.A. Rotea [48],[49]. As shown in Figure 6.1, the system consists of a mass M which is constrained to translate horizontally. The cart is connected to an inertially fixed point with a linear spring. Mounted on the cart is a "proof body" actuator of mass m and moment of inertia I. The proof body rotates relatively to the vertical line passing through the cart mass center. A horizontal force F acting on the cart which is regarded as an external disturbance force. A motor on the cart can be used to generate a torque N to control the proof mass in such a way that the force F has minimal effect on the cart's position. In other words, the external force F will be attenuated as much as possible on the cart by appropriately choosing the control input torque N. The nonlinearity of the problem comes from the interaction between the translational motion of the cart and the rotational motion of the eccentric proof mass.

The equations of motion , with certain normalization, for this nonlinear system is in the form:

$$\ddot{\xi} + \xi = \epsilon(\dot{\theta}^2 sin\theta - \ddot{\theta} cos\theta) + w \tag{6.1}$$

$$\ddot{\theta} = -\epsilon\ddot{\xi} cos\theta + u \tag{6.2}$$

where ξ is the (non-dimensionalized) displacement of the cart and θ is the angular position of the proof body. w and u are the (non-dimensionalized) disturbance and control inputs, respectively. The coupling between the translational and rotational motions is captured by the parameter ϵ which is defined by

$$\epsilon = \frac{me}{\sqrt{(I + me^2)(M + m)}} \tag{6.3}$$

where e is the eccentricity of the proof body. Clearly, $0 \leq \epsilon < 1$ and $\epsilon = 0$ if and only if $e = 0$; in the case, the translational and rotational motions decouple and equations (6.1) and (6.2) reduce to

$$\begin{aligned} \ddot{\xi} + \xi &= w \\ \ddot{\theta} &= u \end{aligned}$$

Letting $x = [x_1\ x_2\ x_3\ x_4]^T = [\xi\ \dot{\xi}\ \theta\ \dot{\theta}]^T$, the state space representation of the system can be written as

$$\begin{aligned} \dot{x} &= f(x) + g_1(x)u + g_2(x)w \\ y &= x_1 \end{aligned}$$

where

$$f = \begin{bmatrix} x_2 \\ \frac{-x_1+\epsilon x_4^2 sinx_3}{1-\epsilon^2 cos^2 x_3} \\ x_4 \\ \frac{\epsilon cosx_3(x_1-\epsilon x_4^2 sinx_3)}{1-\epsilon^2 cos^2 x_3} \end{bmatrix}, g_1 = \begin{bmatrix} 0 \\ \frac{-\epsilon cosx_3}{1-\epsilon^2 cos^2 x_3} \\ 0 \\ \frac{1}{1-\epsilon^2 cos^2 x_3} \end{bmatrix}, g_2 == \begin{bmatrix} 0 \\ \frac{1}{1-\epsilon^2 cos^2 x_3} \\ 0 \\ \frac{-\epsilon cosx_3}{1-\epsilon^2 cos^2 x_3} \end{bmatrix} \tag{6.4}$$

with $1 - \epsilon^2 cos^2 x_3 \neq 0$ for all x_3 and $\epsilon < 1$.

The objective of this problem is to design a state feedback control law such that, under a sinusoidal disturbance $w(t) = Asin\omega t$, the closed loop system is asymptotically stable, and the position of the cart can asymptotically approach 0. That is,

$$\lim_{t\to\infty} y(t) \ = \ 0$$

This problem has been called nonlinear benchmark problem and has attracted a lot of attention in nonlinear control community since 1995. The problem is interesting since the system is nonminimum phase, and hence posed a great challenge to existing control methods. In the following, we will first show that the system with w disconnected is a nonminimum phase system. To this end, using the approach described in [33], we can derive the zero dynamics of the system with $w = 0$ as follows

$$\dot{x}_3 \ = \ x_4 \tag{6.5}$$

$$\dot{x}_4 \ = \ x_4^2 tanx_3 \tag{6.6}$$

Clearly, the zero dynamics is unstable. Next we will formulate the above problem as an output regulation problem. For this purpose, we also need to model the disturbance by the following exosystem $\dot{v} = Sv$ where

$$v \ = \ \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}, \ S = \begin{bmatrix} 0 & \omega \\ -\omega & 0 \end{bmatrix}, \ v_0 = \begin{bmatrix} 0 \\ A \end{bmatrix}$$

which clearly yields $w(t) = v_1(t)$. Next, we need to derive the center manifold equations. To this end, we first obtain the zero dynamics of the augmented systems consisting of the plant and the exosystem as follows:

$$\begin{aligned} \dot{x}_3 &= x_4 \\ \dot{x}_4 &= x_4^2 tanx_3 + \frac{v_1}{\epsilon} secx_3 \\ \dot{v} &= Sv \end{aligned}$$

Clearly, from Chapter 3, the center manifold equations corresponding to the above equation is

$$\begin{aligned}\frac{\partial \mathbf{x}_3(v)}{\partial v} Sv &= \mathbf{x}_4(v) \\ \frac{\partial \mathbf{x}_4(v)}{\partial v} Sv &= \mathbf{x}_4^2 tan\mathbf{x}_3(v) + \frac{v_1}{\epsilon} sec\mathbf{x}_3(v)\end{aligned} \tag{6.7}$$

Once we obtain the solution of the center manifold equations, then the solution of the regulator equations are in the form:

$$\mathbf{x}(v) = \begin{bmatrix} 0 \\ 0 \\ \mathbf{x}_3(v) \\ \mathbf{x}_4(v) \end{bmatrix}$$

and

$$\mathbf{u}(v) = \mathbf{x}_4^2(v) tan\mathbf{x}_3(v) + \frac{v_1}{\epsilon cos\mathbf{x}_3(v)}$$

Again, it is practically impossible for obtaining an exact solutions for these center manifold equations due to its nonlinear nature. However, we propose a new method that the approximated solution of the center manifold equations will be obtained by using neural based approach. we use a three layer feedforward neural network model, similar to Figure 3.1, where the inputs come from the exosystem. Note that both equations consist of two unknown functions $\mathbf{x}_3(v)$ and $\mathbf{x}_4(v)$, hence the outputs of the network model are the approximated solutions of $\mathbf{x}_3(v)$ and $\mathbf{x}_4(v)$.

6.2 Neural based Approximation of the Center Manifold Equations of TORA System

The mathematical expression for the outputs of the neural network are shown below:

$$\hat{\mathbf{x}}_3(W, v) = \sum_{i=1}^{N} w_i^{O1} \phi(y_i) \tag{6.8}$$

$$\hat{\mathbf{x}}_4(W, v) = \sum_{i=1}^{N} w_i^{O2} \phi(y_i) \tag{6.9}$$

where

$$y_i = \sum_{j=1}^{2} w_{ij}^I v_i + w_{i0}^I \tag{6.10}$$

and N is number of hidden neurons, and $\phi(.)$ is the hypertangent activation function. $\hat{\mathbf{x}}_3(W,v)$ and $\hat{\mathbf{x}}_4(W,v)$ are the approximated solutions of $\mathbf{x}_3(v)$ and $\mathbf{x}_4(v)$ respectively. An objective function, defined by the half sum of square of equations (6.8) and (6.9), is minimized to the desired small value by adjusting the values of weights. For simplicity, we will use $\hat{\mathbf{x}}_3$ and $\hat{\mathbf{x}}_4$ to represent the outputs of the neural network.

$$e_1 = \frac{\partial \hat{\mathbf{x}}_3}{\partial v} Sv - \hat{\mathbf{x}}_4 \tag{6.11}$$

$$e_2 = \frac{\partial \hat{\mathbf{x}}_4}{\partial v} Sv - \hat{\mathbf{x}}_4^2 tan\hat{\mathbf{x}}_3 - \frac{v_1}{\epsilon} sec\hat{\mathbf{x}}_3 \tag{6.12}$$

$$J(W,v) = \frac{1}{2}(e_1^2 + e_2^2) \tag{6.13}$$

$$Q(W) = \sum_{v \in \Gamma_d} J(W,v) \tag{6.14}$$

The bias terms of the hidden neurons are all put as 0 which performs a faster rate of optimization of equation (6.14). In addition, the necessary requirement $\mathbf{x}(0) = 0$ will be satisfied. We will discuss the existence of the bias term later. The error function, as shown in equation (6.13), $\tilde{J}^i(v)$ with respect to the Taylor approximated solution of $\tilde{\mathbf{x}}^i(v)$ are represented as follows:

$$\begin{aligned} e_1^i &= \frac{\partial \tilde{\mathbf{x}}_3^i(v)}{\partial v} Sv - \tilde{\mathbf{x}}_4^i(v) \\ e_2^i &= \frac{\partial \tilde{\mathbf{x}}_4^i(v)}{\partial v} Sv - \tilde{\mathbf{x}}_4^i(v)^2 tan\tilde{\mathbf{x}}_3^i(v) - \frac{v_1}{\epsilon} sec\tilde{\mathbf{x}}_3^i(v) \\ J^i(v) &= \frac{1}{2}({e_1^i}^2 + {e_2^i}^2) \end{aligned}$$

The detailed derivations for the gradient with respect to weights are shown as follows:

$$\frac{\partial Q(W)}{\partial w_i^{O1}} = \sum_{p=1}^{T_p} [e_{1p}(\frac{\partial e_{1p}}{\partial w_i^{O1}}) + e_{2p}(\frac{\partial e_{2p}}{\partial w_i^{O1}})] \tag{6.15}$$

$$\frac{\partial Q(W)}{\partial w_i^{O2}} = \sum_{p=1}^{T_p} [e_{1p}(\frac{\partial e_{1p}}{\partial w_i^{O2}}) + e_{2p}(\frac{\partial e_{2p}}{\partial w_{ij}^{O2}})] \tag{6.16}$$

$$\frac{\partial Q(W)}{\partial w_{ij}^{I}} = \sum_{p=1}^{T_p} [e_{1p}(\frac{\partial e_{1p}}{\partial w_{ij}^{I}}) + e_{2p}(\frac{\partial e_{2p}}{\partial w_{ij}^{I}})] \tag{6.17}$$

where

$$\frac{\partial e_{1p}}{\partial w_i^{O1}} = \phi'(y_i)(w_{i1}^I \omega v_2 - w_{i2}^I \omega v_1) \tag{6.18}$$

$$\frac{\partial e_{2p}}{\partial w_i^{O1}} = -\phi(y_i)(\hat{\mathbf{x}}_{4p}^2 sec^2\hat{\mathbf{x}}_{3p} + \frac{v_1}{\epsilon} sec\hat{\mathbf{x}}_{3p} tan\hat{\mathbf{x}}_{3p}) \tag{6.19}$$

$$\frac{\partial e_{1p}}{\partial w_i^{O2}} = -\phi(y_i) \tag{6.20}$$

$$\frac{\partial e_{2p}}{\partial w_i^{O2}} = \phi'(y_i)(w_{i1}^I \omega v_2 - w_{i2}^I \omega v_1) - 2\hat{\mathbf{x}}_{4p} tan\hat{\mathbf{x}}_{3p}\phi(y_i) \tag{6.21}$$

$$\begin{aligned}\frac{\partial e_{1p}}{\partial w_{i1}^I} &= -2w_i^{O1}\phi(y_i)\phi'(y_i)(w_{i1}^I \omega v_2 - w_{i2}^I \omega v_1)v_1 + \phi'(y_i)w_i^{O1}\omega v_2 \\ &\quad -\phi'(y_i)w_i^{O2}v_1\end{aligned} \tag{6.22}$$

$$\begin{aligned}\frac{\partial e_{1p}}{\partial w_{i2}^I} &= -2w_i^{O1}\phi(y_i)\phi'(y_i)(w_{i1}^I \omega v_2 - w_{i2}^I \omega v_1)v_2 - \phi'(y_i)w_i^{O1}\omega v_1 \\ &\quad -\phi'(y_i)w_i^{O2}v_2\end{aligned} \tag{6.23}$$

$$\begin{aligned}\frac{\partial e_{2p}}{\partial w_{i1}^I} &= -2w_i^{O2}\phi(y_i)\phi'(y_i)(w_{i1}^I \omega v_2 - w_{i2}^I \omega v_1)v_1 + \phi'(y_i)w_i^{O2}\omega v_2 \\ &\quad -2\hat{\mathbf{x}}_{4p} tan\hat{\mathbf{x}}_{3p}\phi'(y_i)w_i^{O2}v_1 \\ &\quad -\phi'(y_i)w_i^{O1}v_1(\hat{\mathbf{x}}_{4p}^2 sec^2\hat{\mathbf{x}}_{3p} + \frac{v_1}{\epsilon} sec\hat{\mathbf{x}}_{3p} tan\hat{\mathbf{x}}_{3p})\end{aligned} \tag{6.24}$$

$$\begin{aligned}\frac{\partial e_{2p}}{\partial w_{i2}^I} &= -2w_i^{O2}\phi(y_i)\phi'(y_i)(w_{i1}^I \omega v_2 - w_{i2}^I \omega v_1)v_2 - \phi'(y_i)w_i^{O2}\omega v_1 \\ &\quad -2\hat{\mathbf{x}}_{4p} tan\hat{\mathbf{x}}_{3p}\phi'(y_i)w_i^{O2}v_2 \\ &\quad -\phi'(y_i)w_i^{O1}v_2(\hat{\mathbf{x}}_{4p}^2 sec^2\hat{\mathbf{x}}_{3p} + \frac{v_1}{\epsilon} sec\hat{\mathbf{x}}_{3p} tan\hat{\mathbf{x}}_{3p})\end{aligned} \tag{6.25}$$

Choosing suitable N hidden neurons and a set of training pattern in a defined region $||v|| \leq 2$, the objective function $Q(W)$ has been minimized to about 10^{-3}, where $\Gamma = \{v \in R^2 \mid ||v|| \leq 2\}$, and $\Gamma_d = \{(\rho sin\theta, \rho cos\theta) \mid \rho = 0.2, 0.4, \cdots 2, \theta = -\pi, -\frac{5}{6}\pi, -\frac{4}{6}\pi \cdots, 0, \frac{1}{6}\pi \cdots, \frac{5}{6}\pi\}$.

6.3 Simulation Results

On the other hand, Taylor series approximation method is proposed by J.Huang in [33] where the approximated series of solution (6.7) and the control law are in the form of

$$\tilde{\mathbf{x}}(v) = \begin{bmatrix} 0 \\ 0 \\ a_1 v_1 + a_{12} v_1 v_2^2 + a_{30} v_1^3 + \cdots \\ b_2 v_2 + b_{21} v_1^2 v_2 + b_{03} v_2^3 + \cdots \end{bmatrix} \tag{6.26}$$

and the Taylor approximated solution of the control law is

$$\tilde{\mathbf{u}}(v) = \frac{v_1}{\epsilon} + a_1 b_2^2 v_1 v_2^2 + \frac{a_1^2 v_2^3}{2\epsilon} + \cdots \tag{6.27}$$

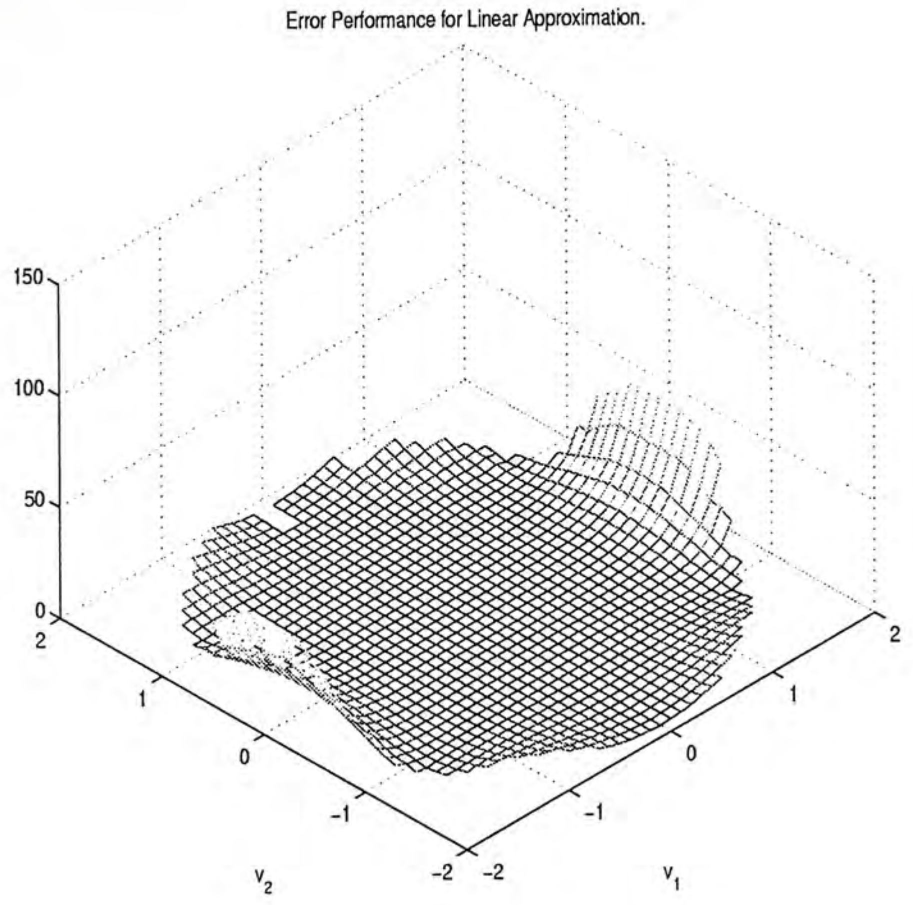


Figure 6.2: The Performance Surfaces of Linear Approximation $Q^{[1]}(v)$

where a_1, b_2, b_{21}, b_{03}, a_{12} and a_{30} are the coefficients of the polynomials. That is,

$$a_1 = -\tfrac{1}{\epsilon\omega^2}, \qquad b_2 = -\tfrac{1}{\epsilon\omega}$$
$$b_{21} = -\tfrac{a_1^2}{2\epsilon\omega}, \quad b_{03} = \tfrac{2b_{21}\omega - a_1 b_2^2}{3\omega}$$
$$a_{12} = \tfrac{b_{03}}{\omega}, \qquad a_{30} = \tfrac{b_{21}+2\omega a_{12}}{3\omega}$$

Since we are going to compare the results from our proposed approach with the Taylor series approximated result of $i = 1, 3$, we thus only show the coefficients of the series within order equal or below 3 where

$$\tilde{\mathbf{x}}^1(v) = \begin{bmatrix} 0 \\ 0 \\ a_1 v_1 \\ b_2 v_2 \end{bmatrix} \tag{6.28}$$

$$\tilde{\mathbf{x}}^3(v) = \begin{bmatrix} 0 \\ 0 \\ a_1 v_1 + a_{12} v_1 v_2^2 + a_{30} v_1^3 \\ b_2 v_2 + b_{21} v_1^2 v_2 + b_{03} v_2^3 \end{bmatrix} \tag{6.29}$$

$$\tilde{\mathbf{u}}^1(v) = \frac{v_1}{\epsilon} \tag{6.30}$$

$$\tilde{\mathbf{u}}^3(v) = \frac{v_1}{\epsilon} + a_1 b_2^2 v_1 v_2^2 + \frac{a_1^2 v_2^3}{2\epsilon} \tag{6.31}$$

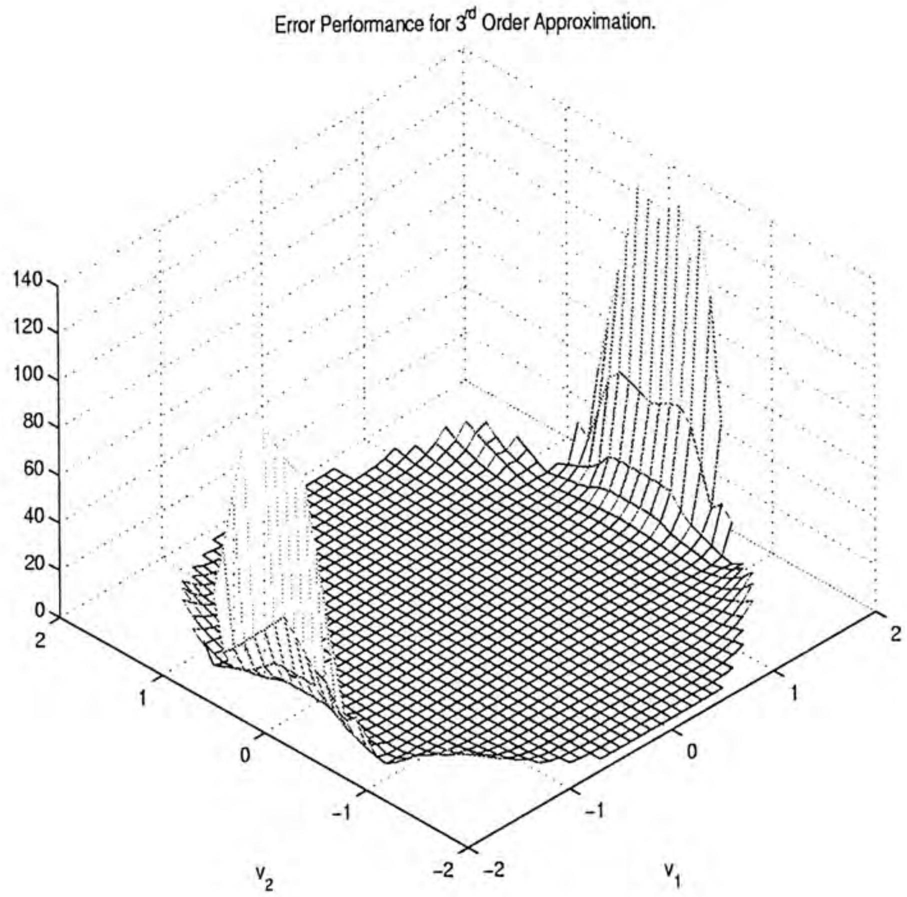


Figure 6.3: The Performance Surfaces of 3^{rd} order Approximation $Q^{[3]}(v)$

Figures 6.2 to 6.4 are the performance surfaces with respect to linear, 3^{rd} order and neural approximation respectively. The performance increases significantly at the outer region of the surface, however, the largest performance value of the neural based approximation is the smaller than other two performance surfaces. The roughness of the linearized case performance surface appears obviously in the inner region, however, it is quite hard to distinguish which one gets a better performance between the $3^r d$ case and our neural based case. We take a comparison of the system responses from these approximated results. Similar to the previous example, we have to design the control law like (4.8). For Taylor series approximated approach, the control law is of the form

$$u = \tilde{\mathbf{u}}^i(v) + K[x - \tilde{\mathbf{x}}^i(v)]$$

which constitutes the series of equations (6.26) and (6.27). However, the neural based control law relies only on the outputs of the neural network since $\hat{\mathbf{u}}(W, v)$ is a function of $\hat{\mathbf{x}}_3(W, v)$ and $\hat{\mathbf{x}}_4(W, v)$. In addition, the feedback gain K from $\frac{\partial f(0)}{\partial x} + g(0)K$ is necessarily defined. The poles are chosen from Bessel prototype, that is, $s = (-0.3944 \pm$

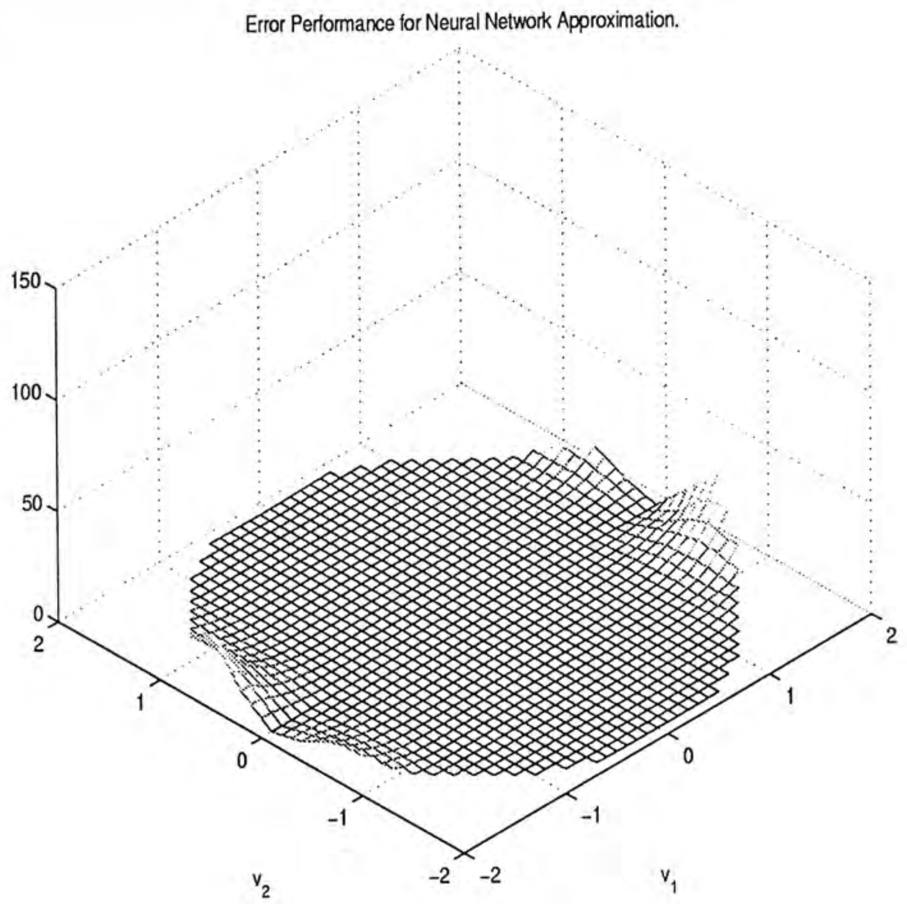


Figure 6.4: The Performance Surfaces of Neural Approximation $Q(W)$

$0.4992i)$, $(-0.5428\pm0.1627i)$ gives the feedback gain $K = [-1.9705\ -5.6723\ 0.1248\ 0.6650]$. The simulation results, with initial condition $x = [0\ 0\ 0\ 0]^T$ are shown here: In order to

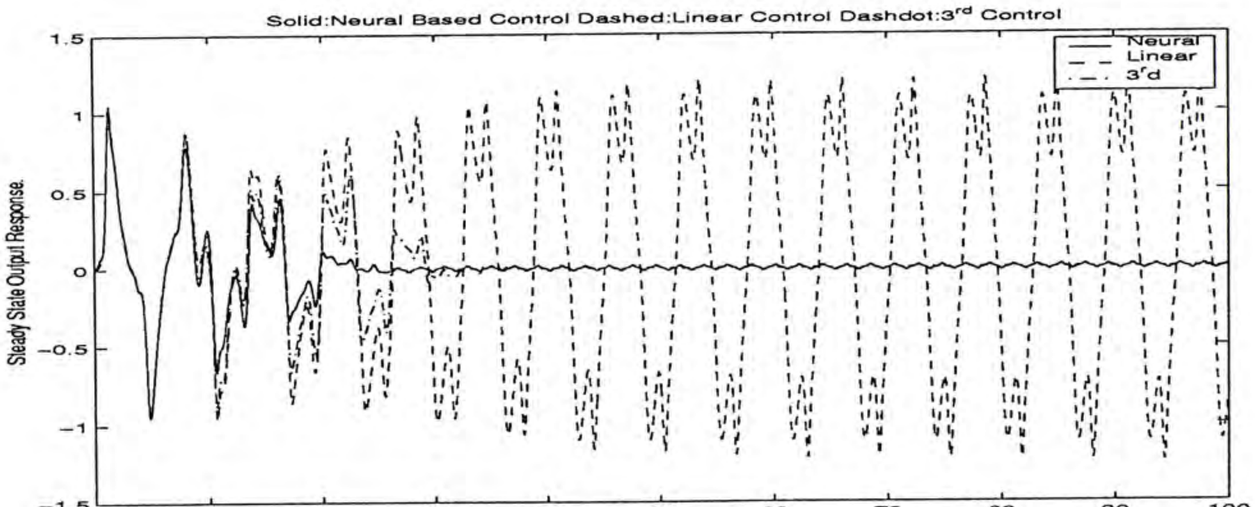


Figure 6.5: Output Response with $\omega = 3$ and $A = 1.8$

make it clearer to compare the performances from these approaches, we take a look at the steady state response.

Figures 6.6 to 6.12 are the simulation results of the system response for three approximation approaches. It is evident that the output of the system from neural based approach

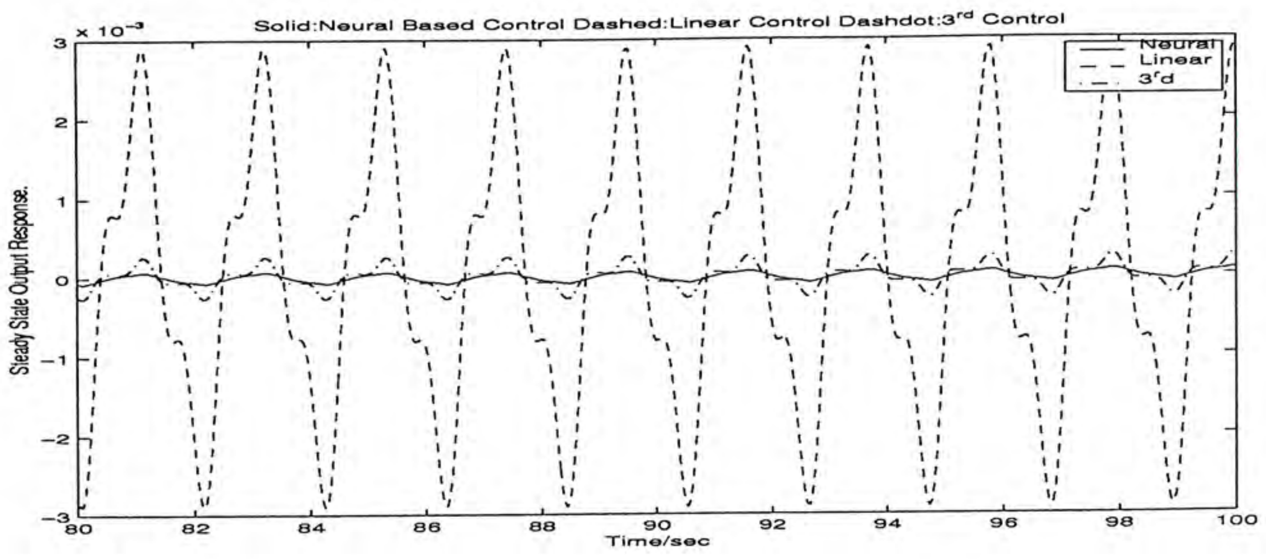


Figure 6.6: Steady State Output Response with $\omega = 3$ and $A = 0.8$

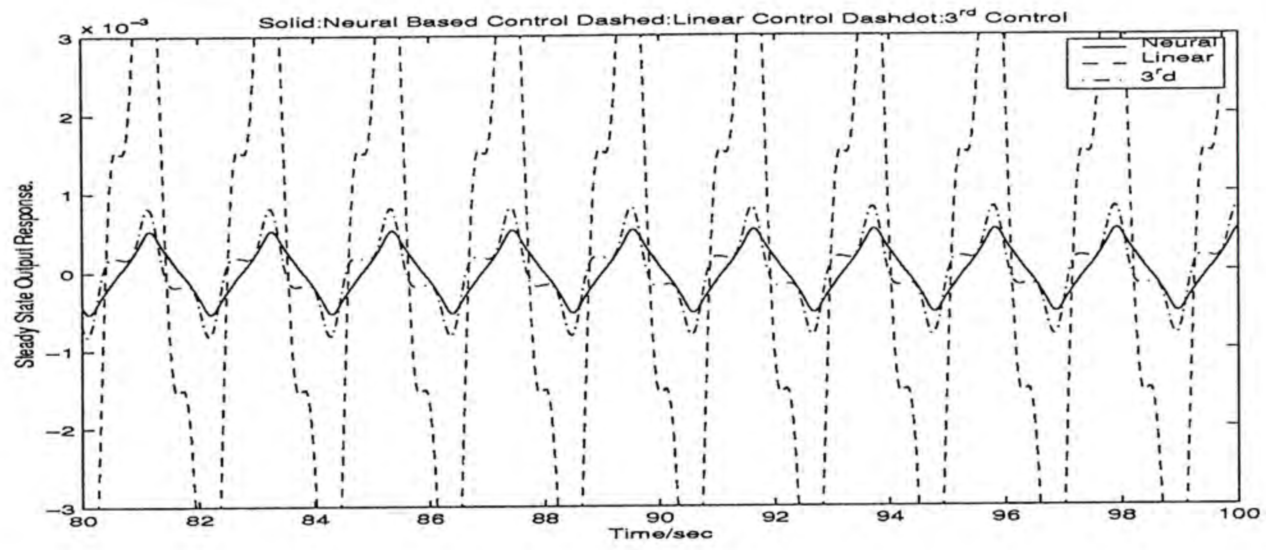


Figure 6.7: Steady State Output Response with $\omega = 3$ and $A = 1.0$

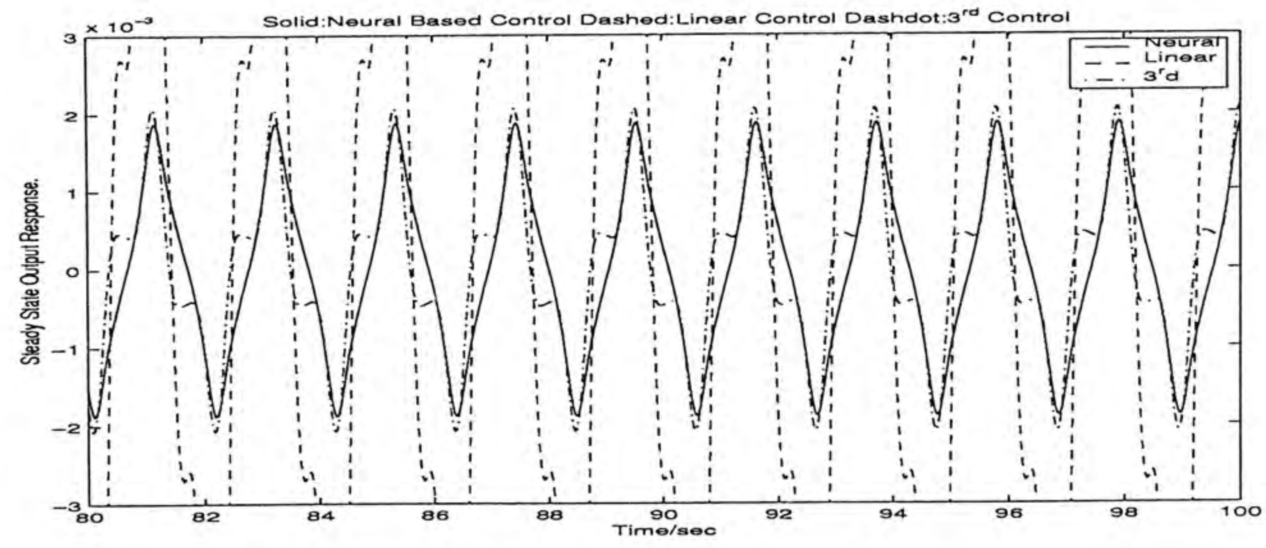


Figure 6.8: Steady State Output Response with $\omega = 3$ and $A = 1.2$

has far smaller steady state error than that from linearized approach but it is only close to the performance from 3^{rd} order Taylor series approximated approach. It does not perform

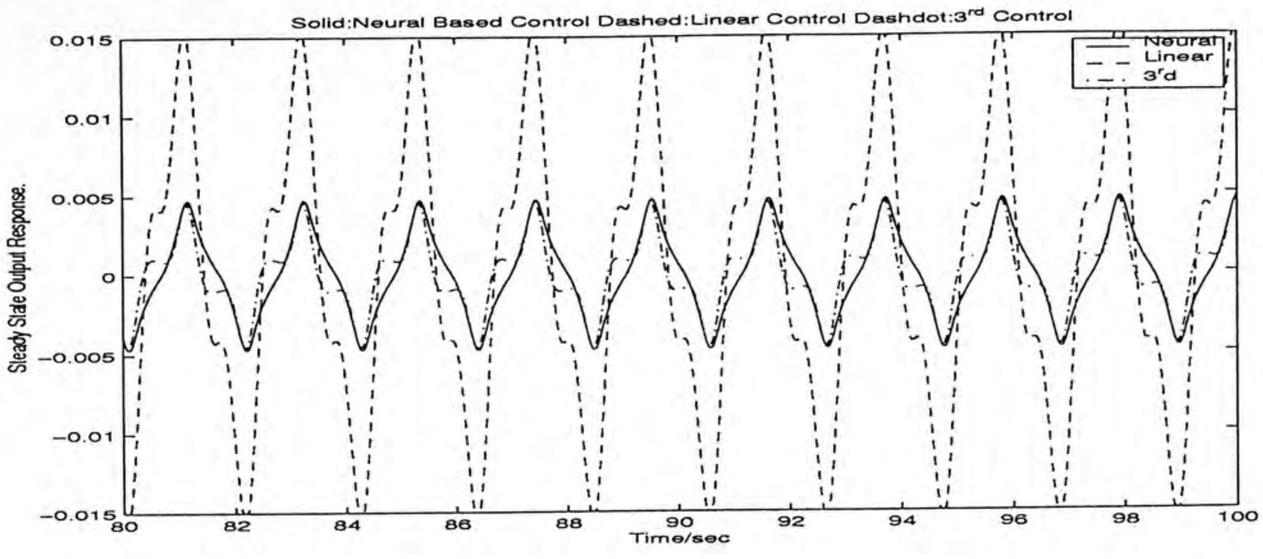


Figure 6.9: Steady State Output Response with $\omega = 3$ and $A = 1.4$

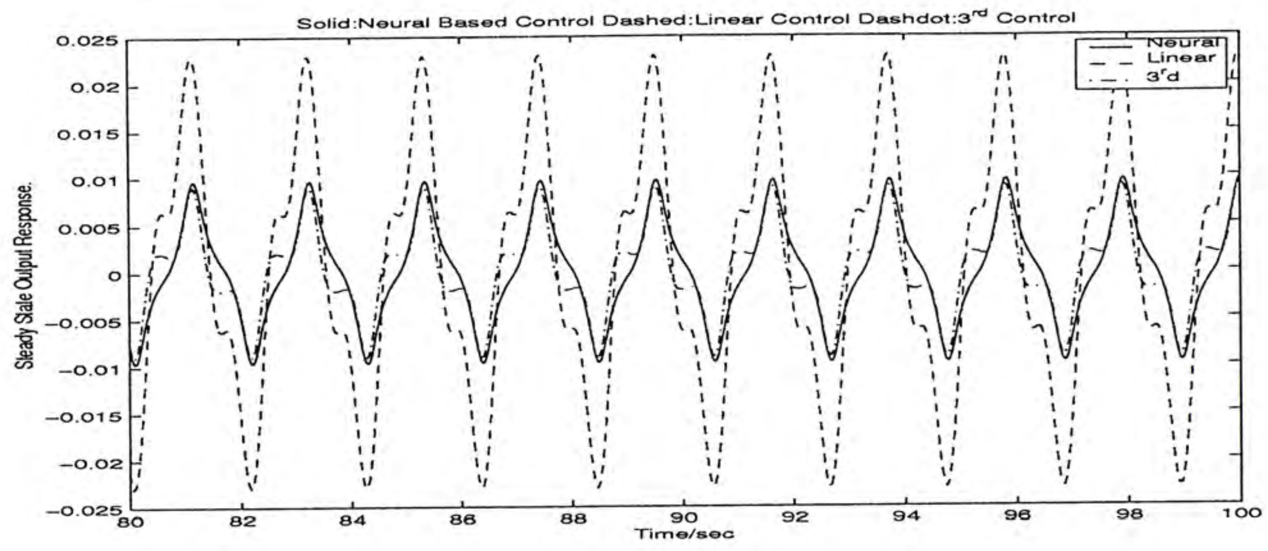


Figure 6.10: Steady State Output Response with $\omega = 3$ and $A = 1.6$

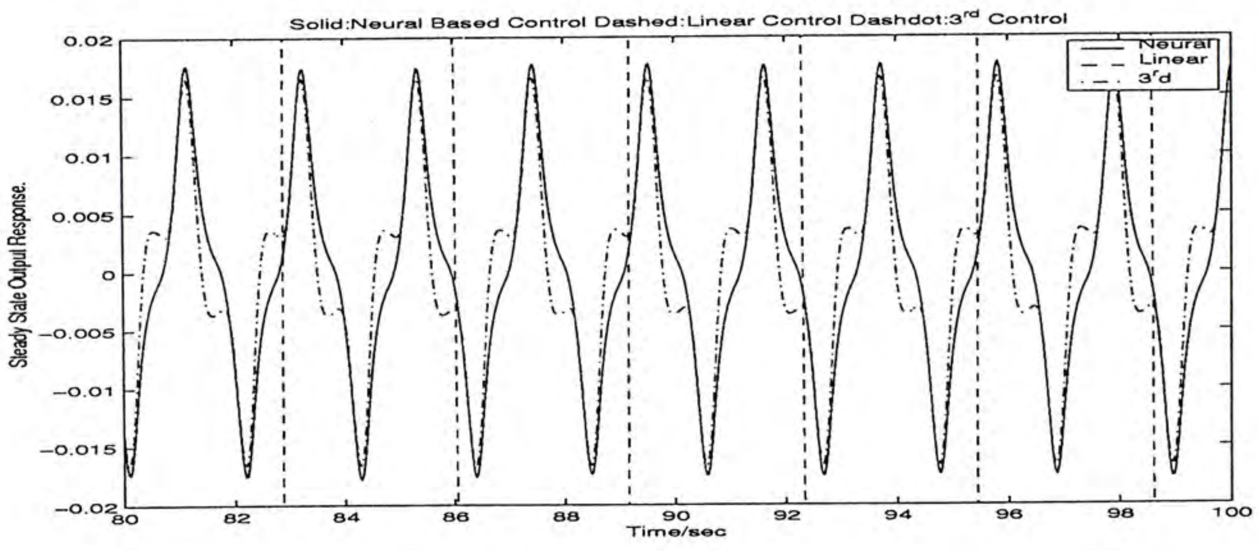


Figure 6.11: Steady State Output Response with $\omega = 3$ and $A = 1.8$

like the example of ball and beam system in the previous chapter. By the way, we will discuss this specific phenomenon in the next section.

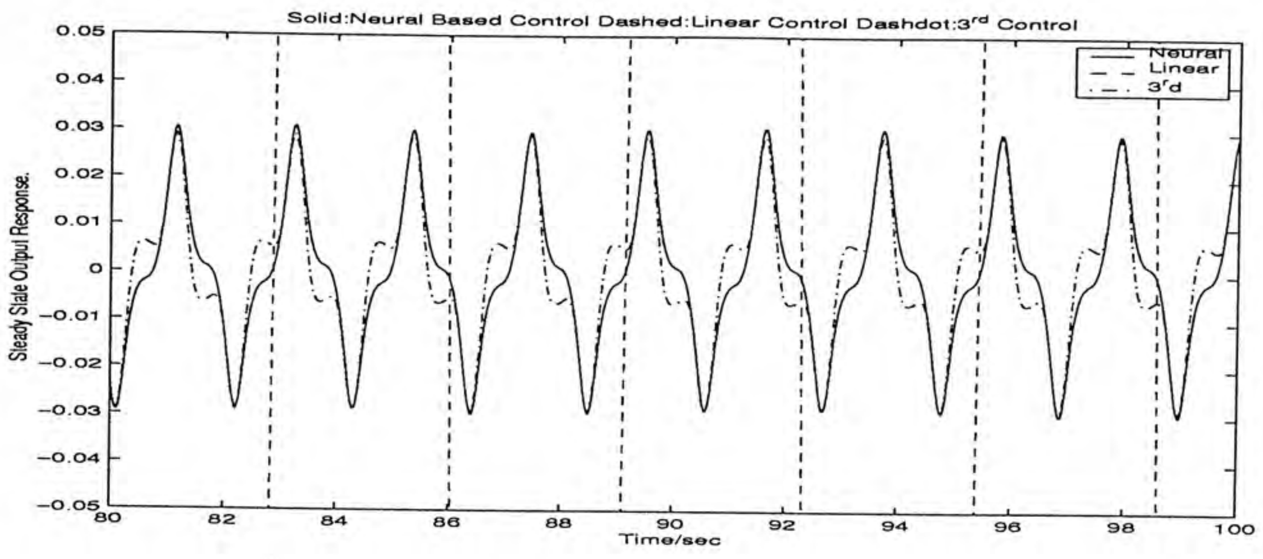


Figure 6.12: Steady State Output Response with $\omega = 3$ and $A = 2$

Amplitude	Linear ITAE	3^{rd} ITAE	Neural ITAE
A=0.8	$\sim 3 \times 10^{-3}$	$\sim 2.5 \times 10^{-4}$	$< 1 \times 10^{-4}$
A=1.0	$\sim 5.5 \times 10^{-3}$	$\sim 7.5 \times 10^{-4}$	$\sim 5 \times 10^{-4}$
A=1.2	$\sim 9.5 \times 10^{-3}$	$\sim 2.1 \times 10^{-3}$	$\sim 1.9 \times 10^{-3}$
A=1.4	$\sim 1.6 \times 10^{-2}$	$\sim 4.5 \times 10^{-3}$	$\sim 4.7 \times 10^{-3}$
A=1.6	$\sim 2.3 \times 10^{-2}$	$\sim 9 \times 10^{-3}$	$\sim 9.7 \times 10^{-3}$
A=1.8	~ 0.9	$\sim 1.65 \times 10^{-2}$	$\sim 1.75 \times 10^{-2}$
A=2.0	~ 1.1	$\sim 2.8 \times 10^{-2}$	$\sim 3 \times 10^{-2}$

Table 6.1: Estimated Steady State Output Repsonse

6.4 Discussion

It is questioned for the effectiveness of the universal approximation theorem that, from the above example, the implementation of neural based approach is not better than 3^{rd} order Taylor series approximated approach as the the objective function $Q(W)$ has been minimized to about 10^{-3}. Probably, it may give a better performance than 3^{rd} order case if $Q(W)$ can be minimized to smaller than 10^{-3}. Unfortunately, during the training process, it can not be minimized more though it has still been taken for a longer time. Therefore, we will consider two design scenarios, namely, 1) the amplitude A of the sinusoidal reference input is allowed to change between $0 \leq A \leq M$ for some fixed constant M, and 2) the amplitude A of the sinusoidal reference input is known and equal to some given constant M. For the first scenario, we need to obtain the approximate solution of the regulator equation in a sphere $\Gamma = \{v \mid ||v||^2 \leq M^2\}$ while for the later we only need to obtain the approximate solution of the regulator equation in a circumference $\Gamma = \{v \mid ||v||^2 = M^2\}$. The first scenario has been described before, we do not need to repeat. To analyze the result of the second scenario, we take $M = 2$, $N = 15$ and discretize Γ to obtain

$$\hat{\Gamma} = \{v_1 = Asin\theta,\ v_2 = Acos\theta \mid A = 2,\ \ \theta = \frac{\pi}{6}j, j = -6, -5, \ldots, 4, 5\} \tag{6.32}$$

and the performance function can be minimized to 1×10^{-4}. Then comparing with figure

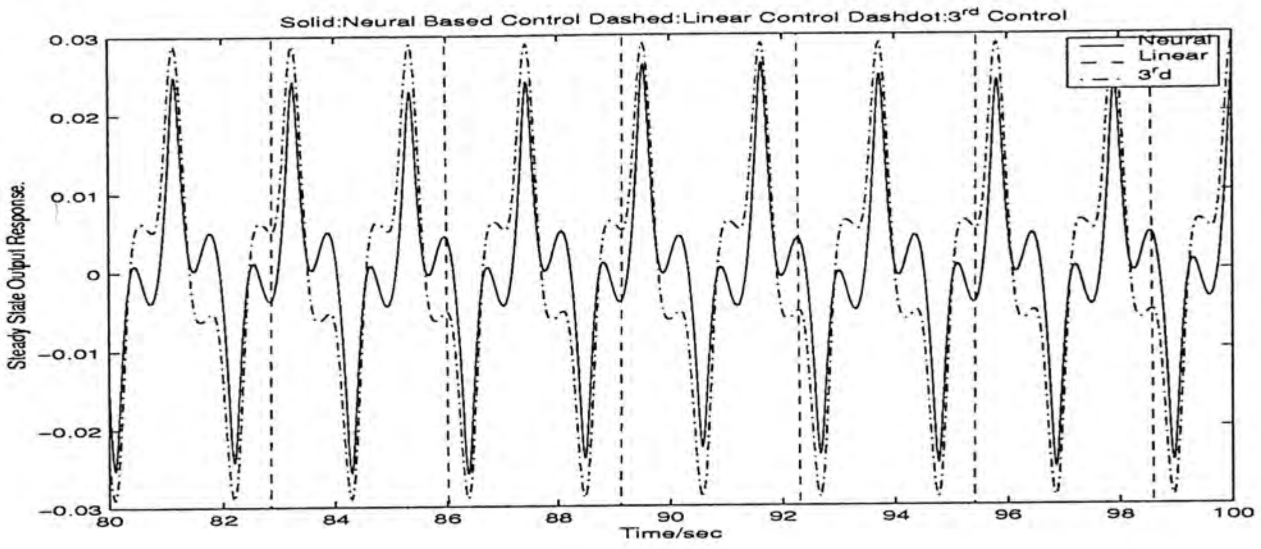


Figure 6.13: Steady State Output Response with $\omega = 3$ and $A = 2$ for Single Circumference Training

6.12, we obtain a weight vector $\hat{W}$ that gives a better system response than 3^{rd} order case in figure 6.13. Actually, there are some limitations of using the feedforward neural network, with one hidden layer, on the functions approximation. Barron established some properties on a three layer feedforward neural network approximation [3]. There are two main factors affecting the training result:

- The accuracy of best approximation is satisfied if the size of the hidden layer m is large enough in accordance with the universal approximation theorem. In other words, we should provide sufficient number of hidden neurons if the problem is more complicated.
- The accuracy of empirical fit to the approximation is satisfied if the ratio of the size

of hidden layer m and the size of training sample N is as small as possible. However, it will be implemented in conflict with the first requirement.

In order to optimize the trade-offs between these two contradiction, the minimization of the performance function may be affected. The above problem reflects the limitations stated in these conclusion. The universal approximation theorem does not guarantee a three layer feedforward neural network is optimum in the sense of learning time, ease of implementation or generalization.

Chapter 7

Conclusion

The research in this thesis has shown that feedforward neural network can be utilized to solve some nonlinear control problems that cannot be easily handled by conventional approach. It is apparent that a feedforward neural network derives its computing power through its massively parallel distributed structure and its ability to learn. These two information-processing capabilities enable it to solve some complex problems that are currently intractable. Some concluding remarks are in order:

1. The neural network approach provides an effective tool to handle a large class of complex problem which is nonlinear in nature. The solvability of center manifold equations is a typical example.

2. The massively parallel nature of the neural network leads itself to an efficient computational tool. The neural network approach is particularly suitable to those problems involving complex nonlinear equations. The approximate control design of the ball and beam system and the TORA system has illustrated this property.

3. As the nonlinear functions are considered to be solved by feedforward neural network, it will be first converted into parameter optimization. A cost function has been defined first which is optimized by the adaptation of the parameters in the network model, called weights, but gradient based method. The steepest gradient descent method guarantees that the cost function will be optimized iteratively. That is,

$$Q(W(n+1)) \quad < \quad Q(W(n)) \tag{7.1}$$

and the optimum solution is possibly found according to the universal approximation theorem where

$$Q(W^*) \leq Q(W) \tag{7.2}$$

The optimum solution will be found with respect to the optimum weights W^*. However, it is practically unnecessary for searching such optimal weights, in other words, that it is sufficient for obtaining a set of weight values which results in an acceptable small error. As illustrated in all examples, we did not have to find the minima of the cost functions. Rather it suffices to obtain an approximated result that results in sufficiently small tracking error in the control design of nonlinear systems.

7.1 Future Works

As we have implemented the neural based approach on the nonlinear control problem successfully, there are still some areas for improvement.

- First of all, we may further improve the gradient based method to speed up the training since it costs a lot of time on the training process with larger number of training patterns involved.
- It is possible to use different activation function in the hidden layer of the neural network such as radial basis function, and compare it with the hyper-tangent function. Also, it is interesting to consider other types of neural networks such as recurrent neural networks or wavelets.
- A lot of other nonlinear control problems also lead to some nonlinear partial differential equations such as HJI equation from the nonlinear H_∞ control. It might be interesting to adapt the approach developed in this thesis to solve the HJI equation.

Appendix A

Center Manifold Theory

Linearization is usually used to study stability of equilibrium points of an autonomous nonlinear system. It can be seen that linearization fails when the Jacobian matrix, evaluated at the equilibrium point, has some eigenvalues with zero real parts. Center manifold theorem is to study stability of the origin of an autonomous system in the critical case when linearization fails. It is the theorem developed by J.Carr [6] and Consider the autonomous system

$$\dot{x} = f(x) \tag{A.1}$$

where f is a C^r vector field ($r \geq 2$) defined on an open subset U of R^n, $f(0) = 0$. Let

$$F = [\frac{\partial f}{\partial x}]_{x=0} \tag{A.2}$$

Invariant Manifold: A C^r submanifold S of U is said to be locally invariant for (A.1), if for each $x^0 \in S$, there exist $t_1 < 0 < t_2$ such that the integral curve $x(t)$ of (A.1) satisfying $x(0) = x^0$ is such that $x(t) \in S$ for all $t \in (t_1, t_2)$.

In other words, let $x|\eta(x) = 0$ be a submanifold of R^n, where $\eta : R^n \to R^p$, then $x|\eta(x) = 0$ is invariant for (A.1) if the solution of an equation $\{\eta(x) = 0\}$ is said to be an Invariant Manifold for system (A.1) if

$$\eta(x(0)) = 0 \Longrightarrow \eta(x(t)) \equiv 0, \quad \forall t \in [0, t_1) \subset R \tag{A.3}$$

where $[0, t_1)$ is any time interval over which the solution $x(t)$ is defined.

Assumption: Consider the case in which the matrix F has all eigenvalues with non-positive real part.

$$\dot{x} = f(x) = Fx + \hat{f}(x) \tag{A.4}$$

we can always find a similarity transformation to reduce F to a block diagonal form:

$$T^{-1}FT = \begin{bmatrix} A & 0 \\ 0 & B \end{bmatrix} \tag{A.5}$$

by means of a linear change of coordinates in U such that the system (A.1) is represented in the form

$$\dot{y} = Ay + g_1(y,z) \tag{A.6}$$

$$\dot{z} = Bz + g_2(y,z) \tag{A.7}$$

where A is an matrix having all eigenvalues with negative real part, B is an matrix having all eigenvalues with zero real part, and the functions g_1, g_2 are C^r functions satisfying

$$g_i(0,0) = 0; \quad \frac{\partial g_i}{\partial y}(0,0) = 0; \quad \frac{\partial g_i}{\partial z}(0,0) = 0 \tag{A.8}$$

for $i = 1, 2$.

Theorem: Suppose the matrix F has n^0 eigenvalues with zero real part, n^- eigenvalues with negative real part. There exist a neighborhood $V \subset R^{n^0}$ of $z = 0$ and a C^r mapping $\pi : V \longrightarrow R^{n^-}$ such that

$$S = \{(y,z) \in (R^{n^-}) \times V : y = \pi(z)\} \tag{A.9}$$

is an invariant manifold for (A.6,A.7), i.e. $y = \pi(z)$ satisfies

$$\frac{\partial \pi}{\partial z}(Bz + g_2(\pi(z),z)) = A\pi(z) + g_1(\pi(z),z) \tag{A.10}$$

and

$$\pi(0) = 0, \quad \frac{\partial \pi}{\partial z}(0) = 0 \tag{A.11}$$

Remark: The manifold defined by (A.9) is called a center manifold [6].

Appendix B

Relation between Center Manifold Equation and Output Regulation Problem

Considering the following system:

$$\dot{x}(t) = f(x(t), v(t)), \quad t \geq 0 \tag{B.1}$$

$$\dot{v}(t) = a(v(t)) \tag{B.2}$$

where $x(t) \in R^n$, $v(t) \in R^q$. f and a are C^2, satisfying $f(0,0) = 0$ and $a(0,0) = 0$.

Suppose none of the eigenvalues of the matrix

$$\frac{\partial f}{\partial x}(0,0) \tag{B.3}$$

have zero real parts, and all the eigenvalues of matrix

$$\frac{\partial a}{\partial v}(0) \tag{B.4}$$

have zero real parts. Then by center manifold theorem, for some $\epsilon > 0$, there exists a C^2 function $\mathbf{x} : R^q \to R^n$ with $\mathbf{x}(0) = 0, \mathbf{x}'(0) = 0$ such that, for $\|v\| < \epsilon$,

$$\frac{\partial \mathbf{x}(v)}{\partial v} a(v) = f(\mathbf{x}(v), v) \tag{B.5}$$

Solutions of (B.1) and (B.2) have the following properties:

(Reduction Principle) The zero solution of Equation (B.1) and (B.2) are stable (asymptotically stable) (unstable) iff the zero solution of (B.2) is stable (asymptotically stable) (unstable). This principle is rather important since we can easily determine the stability of the system by reducing it from higher dimension to lower dimension like as equation (B.2).

Lemma Let $(x(t), v(t))$ be a solution of Equation (B.1) and (B.2) with $(x(0), v(0))$ sufficiently small. Let $\mathbf{x}(v)$ be such that Equation (B.5) holds. Then there exist positive constants K and M such that

$$\|x(t) - \mathbf{x}(v(t))\| \leq Me^{-Kt}\|x(0) - \mathbf{x}(v(0))\| \tag{B.6}$$

for all $t \geq 0$.

This Lemma shows that any trajectory of the system (B.1) and (B.2) starting at a point sufficiently close to $(0, 0)$ will converge to the center manifold as t tends to infinity, with exponential decay.

Biography

Chi Fai Ng comes from Hong Kong. He received the B.Eng in 1997 from the Chinese University of Hong Kong, majoring in Mechanical and Automation Engineering. From 1997 to 1998, he worked as an engineer trainee in JOS technical Services, M.C.L. Engineering Co. Ltd., Hong Kong. Since 1998, he has been working towards his master degree in the Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong.

His research interests include nonlinear control theory and neural networks and applications. Research related to his study has been published by the following conference papers:

[1] C.F. Ng and J. Huang, "Solving the Center Manifold Equation by Feedforward Neural Networks", *Second International ICSC Symposium on Neural Computation NC'2000*, Berlin, Germany, May, 2000.

[2] J. Zhao, C.F. Ng, Jin Wang and J. Huang, "Neural Networks Based Asymptotic Tracking for a benchmark Nonlinear System", *The Chinese Control Conference*, Hong Kong, December 2000, Accepted.

Bibliography

[1] M.S. Ahmed and I.A. Tasadduq, "Neural-net controller for nonlinear plants: design approach through linearization", *IEE Proc-Ctrol theory*, Vol. 141, pp. 315-322, 1994.

[2] P. Antsaklis(Eds), "Special Issue on Neural Network in Control Systems", *IEEE Contr. Syst. Mag.*, Vol. 6, pp. 3-87, 1990.

[3] A.R. Barron, "Universal Approximation Bounds for Superpositions of a Sigmoidal Function", *IEEE Trans. on Information Theory*, vol. 39, pp. 930-945, 1993.

[4] R.T. Bupp, D.S. Bernstein and V.T. Coppola, "Experimental implement of integrator backstepping and passive nonlinear controllers on the RTAC testbed", *International Journal of Robust and Nonlinear Control*, Vol. 8, pp. 435-457, 1998.

[5] C.I. Bynes, F. Delli Priscoli, A. Isidori and W. Kang, "Structurally Stable Output Regulation of Nonlinear Systems", *Automatica*, Vol. 33, pp. 369-385, 1997.

[6] J. Carr, "Applications of Center Manifold Theory", Lecture Notes in Control and Information Sciences, Spring-Verlag, New York, 1981.

[7] T. Chen and H. Chen, "Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems", *IEEE Trans. on Neural Networks*, vol. 6, no. 4, pp. 911-917, 1995.

[8] T. Chen and H. Chen, "Approximation capabilities to functions of several variables, nonlinear functionals, and operators by radial basis functions neural networks", *IEEE Trans. on Neural Networks*, vol. 6, no. 4, pp. 904-910, 1995.

[9] T. Chen, H. Chen and R.W. Liu, "Approximation Capability in $C(\bar{\mathbf{R}}^n)$ by multilayer feedforward networks and related problems", *IEEE Trans. on Neural Networks*, vol. 6, no. 1, pp. 25-30, 1995.

[10] Y.C. Chu and J. Huang, "A neural network method for nonlinear servomechanism problem", *IEEE Transactions on Neural Networks*, pp. 1412-1423, Nov. 1999.

[11] G. Cybenko, "Approximation by superpositions of a sigmoidal function", *Mathematics of Control, Signals and Systems*, vol. 2, no. 4, pp. 303-314, 1989.

[12] E.J. Davison, "The robust control of a servomechanism problem for linear time-invariant multivariable systems", *IEEE Transactions on Automatic Control*, Vol. 35, No. 2, pp. 131-140, 1990.

[13] Howard Demuth and Mark Beale, "Neural Network Toolbox for use with MATLAB", User's Guide Version 3.0, 3.4-11 and 5.3-16.

[14] C.A. Desoer and Y.T. Wang, "Linear time-invariant robust servomechanism problem: A self-Contained Exposition," *Control and Dynamic Systems*,Vol. 16, pp. 81-129, 1980.

[15] Terrence L. Fine, "Feedforward Neural Network Mehtodology", *Springer-Verlag*, New York, 1999.

[16] G.F. Franklin,J.D. Powell and A. Emami-Naeini, "Feedback Control of Dynamic Systems", Third edition Addison Wesley.

[17] J. Hauser, S. Sastry and P. Kokotovic, "Nonlinear control via approximate input-output linearization: The ball and beam example", *IEEE Transactions on Automatic Control*, Vol. 37, no. 3, pp. 392-398, 1992.

[18] S. Haykin, "Neural Network", *A comprehensive foundation*, Macmillan, 1995.

[19] Kateřina Hlaváčková, "Feedforward networks with one hidden layer and their rates of approximation", *UKACC International Conference on CONTROL' 98*, 1-4 Sept. 1998, (Conf. Publ. No. 455), vol. 1, pp. 727-732, 1998.

[20] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, pp. 251–257, 1991.

[21] G.B. Huang and H.A. Babri, "General Approximation Theorem on Feedforward Networks", *International Conference on Information, Communications and Signal Processing ICICS' 97*, Singapore, 9-12 September 1997.

[22] J. Huang, "On the Servomechanism and Noninteracting Control Problems for Nonlinear Systems", Ph.D. dissertation, *The Johns Jopkins University*, 1990.

[23] J. Huang, "Editorial of Special Issue: Output Regulation for Nonlinear Systems", *International Journal of Robust and Nonlinear Control*, Vol. 10, No. 5, pp. 321-322, 2000.

[24] J. Huang, "Asymptotic tracking of a non-minimum phase nonlinear system with non-hyperbolic zero dynamics", *IEEE Transactions on Automatic Control*, Vol. 45, No. 6, June, 2000.

[25] J. Huang, "Output regulation of non-hyperbolic nonlinear systems", *IEEE Transactions on Automatic Control*, Vol. 40, No. 8, pp. 1497-1500, 1995.

[26] J. Huang and C.F. Lin, "On a Robust Nonlinear Servomechanism Problem", *IEEE Transactions on Automatic Control*, Vol. 39, No. 7, pp. 1510-1513, 1994.

[27] J. Huang, and W.J. Rugh, "On a nonlinear multivariable servomechanism Problem," *Automatica*, Vol. 26, No. 6, pp. 963-972, 1990.

[28] J. Huang, and W.J. Rugh, "Stabilization on zero-error manifolds and the nonlinear servomechanism Problem, " *IEEE Transactions on Automatic Control*, Vol. 37, No.7, pp. 1009-1013, 1992.

[29] J. Huang, and W.J. Rugh, "An Approximation Method for the Nonlinear Servomechanism Problem", *IEEE Transaction on Automatic Control*, Vol. AC-37, No.9, pp.1395–1398, 1992.

[30] H.J.C. Huijberts, H. Nijmeijer, R.M.A. Willems, "Regulation and controlled synchronization for complex dynamical systems", *International Journal of Robust and nonlinear Control*, Vol. 10, No. 5, pp. 363-377, 2000.

[31] A.Isidori, "Nonlinear Control Systems", Third edition Springer, 1995.

[32] A. Isidori and C.I. Byrnes, "Output regulation of nonlinear systems", *IEEE Transactions on Automatic Control*, Vol. 35, No.2, pp. 131 - 140.

[33] A.Isidori and J.Huang, "Output Regulation of Nonlinear Systems", Tutorial Workshop S-4, *38th IEEE Conference on Decision and Control*, Phoenix, Arizona, Sunday, December 5th, 1999.

[34] L. Jin, P.N. Nikiforuk and M.M. Gupta, "Direct adaptive output tracking control using multilayered neural networks", *IEE-D*, Vol. 140, pp. 393-398, 1993.

[35] H.K. Khalil, "Nonlinear Systems", second edition *Prentice Hall*, 1996.

[36] H.K. Khalil, "On the design of robust servomechanisms for minimum phase nonlinear systems", *International Journal of Robust and Nonlinear Control*, Vol. 10, No. 5, pp. 339-361, 2000.

[37] I.E. Lagaris, A. Likas and D.I. Fotiadis, "Artificial neural networks for solving ordinary and partial differential equations", *IEEE Trans. Neural Networks*, Vol. 9, pp. 987-1000, 1998.

[38] Y. Liguni and H. Sakai, "A nonlinear regulator design in the presence of system uncertainties using multilayered neural networks", *IEEE Trans. Neural Networks*, Vol. 3, pp. 410-417, 1991.

[39] Ching-Fang Lin, "Advanced Control Systems Design", Prentice Hall Series in Advanced Navigation, Guidance, and Control, and Their Applications, 1994.

[40] K.S. Narendra and K. Parthasarathy, "Identification and control of dynamical system using neural networks", *IEEE Contor. Syst. Mag.*, Vol. 2, pp. 4-27, 1990.

[41] C.F. Ng and J. Huang, "Solving the Center Manifold Equation by Feedforward Neural Networks", *Second International ICSC Symposium on Neural Computation NC'2000*, Berlin, Germany, May, 2000.

[42] Y.M. Park, M.S. Choi and K.Y. Lee, "An optimal tracking neruo-controller for nonlinear dynamic systems", *IEEE Trans. Neural Networks*, Vol. 7, pp. 1099-1110, 1996.

[43] D. Psaltis, A. Sideris and A.A. Yamamura, "A Multilayered neural network control",*IEEE Contr. Syst. Mag.*, Vol. 8, pp. 17-21, 1988.

[44] R. Rojas, "Neural Networks", *A systematic introduction, Springer-Verlag*, Berlin, 1996.

[45] G.A. Rovithakis, "Tracking Control of Multi-Input Affine Nonlinear Dynamical Systems with Unknown Nonlinearities Using Dynamical Neural Networks", *IEEE Trans. Syst., Man, Cybern.-Part B*, Vol. 29, pp. 179-189, 1999.

[46] Robert J. Schalkoff, "Artificial Neural Networks", *McGraw-Hill*, 1997.

[47] J.J.E. Slotine and W. Li, "Applied Nonlinear Control",Prentice Hall, 1991.

[48] Panagiotis Tsiotras, Martin Corless and Mario A. Rotea, "An $\mathcal{L}_2$ Disturbance Attenuation Approach to the Nonlinear Benchmark Problem", *Proceedings of the American Control Conference*, Seattle, Washington, FP16 - 3:50, June, 1996..

[49] Panagiotis Tsiotras, Martin Corless and Mario A. Rotea, "An $\mathcal{L}_2$ Disturbance Attenuation Approach to the Nonlinear Benchmark Problem", *International Journal of Robust and Nonlinear Control*, vol. 8, 311-330, 1998.

[50] Jin Wang, J. Huang and S.S.T. Yau, "Approximate Nonlinear Output Regulation based on the Universal Approximation Theorem", *Proceedings of World Multiconference SCI/ISAS'99*, Florida, USA, August, Vol. 7, pp. 218 - 225, 1999.

[51] S.H. Yu and A.M. Annaswamy, "Stable Neural Controllers for Nonlinear Dynamic Systems", *Automatica*, Vol. 34, pp. 641-650, 1999.