National Library of Canada
Acquisitions and Bibliographic Services Branch
395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale du Canada
Direction des acquisitions et des services bibliographiques
395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

Canada

Neural Network based Decentralized Control of an MZSH System

Hossein S. Saboksayr

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of Master of Applied Science at

Concordia University

Montréal, Québec, Canada

August 18 1995

ISBN 0-612-05135-8

Canada

ABSTRACT

Neural Network based Decentralized
Control of an MZSH System

Hossein S. Saboksayr

Efficient operation of space heating systems is a practical control problem of considerable economic significance. In this thesis, a multizone space heating (MZSH) system is considered. The MZSH system consists of a boiler, two environmental zones and two heat pumps (one for each zone) and the associated distribution network. The control problem is to operate the boiler and the heat pumps such that good zone temperature control can be achieved and energy savings can be realized by implementing occupied and unoccupied setpoint changes. This task of combining the setpoint changes initiated by the occupants and those initiated by the supervisory controller such as night setback into one control strategy is investigated.

To this end a multivariable decentralized neural network controller is proposed. The proposed controller is time-varying and its design is based on the minimization of a decentralized cost function. The performance of the designed controller is compared with published results. It is shown that the neural networks are trainable for MZSH systems and the control system performs well over a wide range of operation and gives better disturbance rejection compared to the existing controllers.

To my parents
Batoul Seradj and S. Haydar Saboksayr

ACKNOWLEDGEMENTS

I appreciate my supervisors Prof. R. V. Patel and Dr. M. Zaheeruddin for their inspiration, support and help throughout the span of this thesis.

I would also like to thank Dr. S. A. K. Al-Assadi for providing the initial data and information needed in this thesis and for being helpful and friendly.

I would appreciate N. Ayoub, T. Ponniah, and H. Kovalcik who helped me when there was a system problem. Also, I would like to thank Ms. C. Devan in the department of electrical and computer engineering who has always been kind and helpful.

I appreciate all who have been supportive but their names are not included here.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

1.1 Introduction

Because of the importance of heating systems in northern climates, there have been several control strategies devised for the Heating Ventilating and Air Conditioning (HVAC) systems. A very successful control strategy which is still in use is PID control. The design of such controllers has roots in linear control system design for single-input single-output systems. The simplicity of the concept and the controller structure has allowed such a design method to survive for a long time. However, it is well known that control based on linear models often sacrifices the system performance when applied to nonlinear systems.

In this thesis we consider a particular class of HVAC systems known as Multizone Space Heating (MZSH) systems. The MZSH system consists of a boiler, a heat pump for each zone, and a building with two environmental zones. The MZSH system is a nonlinear system. Furthermore, it undergoes large changes in operating

point during its day-to-day operation such as occupied and unoccupied period operation. For these reasons, the control design based on linear models is not likely to be satisfactory.

In recent years, there has been a move towards multivariable controller design. Such controllers are capable of providing significant improvement in performance. Some attempts have been made in order to simplify multivariable controller structures. A good example of a multivariable controller is that which results from the linear quadratic optimal control approach which tries to minimize a quadratic measure of the system performance (or cost function). However, defining how fast a response should be or how much overshoot or undershoot is tolerable in optimal control problem is difficult. But, the cost function can include a measure of the speed of the response as well as a measure of energy which is spent for the control action. Thus, an energy saving strategy can be appropriately posed in this form of design.

1.1.1 Description of the Problem

In designing controllers for MZSH systems in buildings, two issues are important: good setpoint control and speed of the response. Both of these requirements are necessary to achieve thermal comfort and energy efficiency in buildings [1].

Thermal comfort is affected by outdoor disturbances and zone setpoint changes applied by occupants. The temperature of each room in a building is affected by the outdoor air temperature changes and changes in the solar radiation into each room. So, keeping room temperatures as close to their setpoints as possible in the presence of such disturbances, i.e. disturbance rejection, is essential to achieve thermal comfort. Also, when an occupant changes his/her room temperature setpoint, he/she expects the room temperature to change as fast as possible to its new setpoint. So, fast response of a control system to setpoint (or reference input) changes (regulation) is a measure of comfort as well.

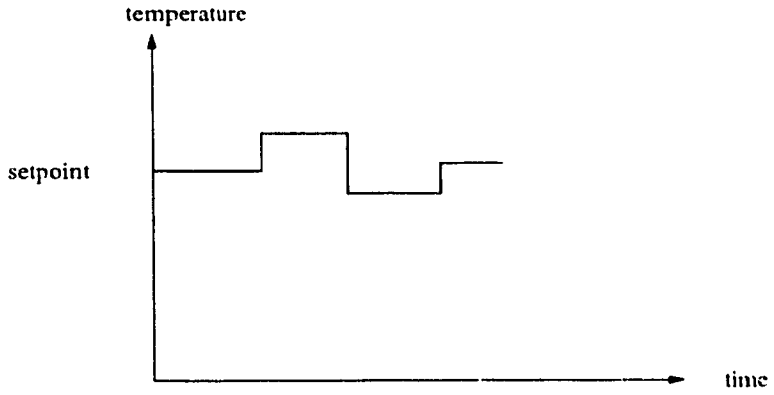

Figure 1.1: Typical setpoint changes applied by occupants.

The goal is to design controllers by incorporating the above features. However, the complexity and nonlinearity of the heating systems have made the problem of finding a general solution difficult. Thus there is significant on-going research in this area.

1.1.2 Comfort and Energy Efficiency

Energy efficiency of controllers for MZSH systems is determined by the energy-saving strategy of the control operation and is taken care of by supervisory controller. The energy saving strategy here generally consists of decreasing the temperature of the boiler and zones (rooms) by few degrees when the building is not occupied.

Although, change of setpoints by a supervisory controller in an energy-saving strategy appears similar to setpoint changes applied by occupants but they represent two different viewpoints. Setpoint changes applied by occupants are usually small changes which are taken care of by regulators (Figure 1.1). Temperature changes applied by supervisory controller are usually large resulting in saturations in control system and causing regulators to not perform optimally (Figure 1.2).

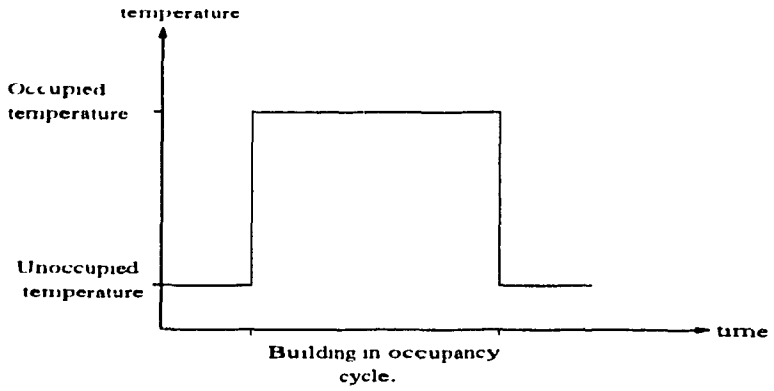

Figure 1.2: Typical setpoint changes applied by a supervisory controller as an energy saving strategy.

The desired response of the control system to occupied/unoccupied setpoint switch should be fast and without causing overshoot or undershoot which means letting the zone temperature remain at its lower level for as long as possible and giving the control system maximum control flexibility (considering limits caused by saturations) to reach the new setpoint.

1.1.3 The Goal

The problem of controlling the multi-zone space heating (MZSH) systems is a nonlinear control problem. Dealing with small as well as large changes in zone setpoints (reference signals) makes the MZSH system hard to control by using linear design methods. In this study the goal is to design a nonlinear time-varying controller which lets us deal with different aspects of the problem together not as several control problems for a set of linear (linearized) models.

For accomplishing this goal neural networks which have shown promising performance in nonlinear problems are employed. The structure of the controller and all design parameters are chosen conservatively in order to be able to deal with the inherent uncertainties involved with neural network training.

1.2 Current Control Strategies for Heating Systems

As was said earlier, PID controllers are widely used for controlling the HVAC systems. These controllers were originally designed for single-input, single-output linear systems. The multivariable nature of heating systems requires a multivariable structure for the controller. Some researchers have attempted to design multivariable controllers which are appropriate for heating systems applications. The nonlinearity of such systems is an issue which makes the known linear design methods inefficient for heating systems. Thus, attempts have been made to improve the performance of existing controllers or to design new multivariable, adaptive or nonlinear controllers.

In the following a brief literature review of some controller design attempts is presented.

Nesler [2] addresses the need for controller parameter changes because of changing condition of HVAC systems is addressed. The evolution of a controller parameter adjustment method is explained. The HVAC system, in this method, are modeled by a first order linear system with a time delay and controller parameters are calculated for the desired characteristic of the control system. The controller parameter design or adjustment used to be accomplished entirely by engineers. This design, or parameter adjustment, procedure was later done by using the assistance of computers. In computer assisted controller tuning the open loop step response data is collected by the operator and a computer program fits a model to the data and the controller parameters are computed. Then, the operator adjusts the controller parameters. The adaptive control scheme was also used in this method in order to remove the need to the data collection by the operator. In automatic controller tuning the adaptive control scheme identifies the parameters of the heating system and finds the controller parameters continuously or periodically. In this scheme a first order model with a delay is assigned to the process and its parameters are calculated by

using the RLS, Recursive Least Square algorithm. Then the controller parameters are computed by using the model. This approach is a single-input single-output approach and required open-loop test data. The tests have to be repeated if the setpoint changes.

Wallenborg [3] introduces a self tuning controller which is based on a discrete design approach. When the controller tuning is required, a pulse train is applied to the input of the heating system. The response of the system is measured at a few discrete instants. A first order discrete time model of the system is found by using the measured data. The controller parameters are computed for the linear model and by using the (frequency domain) RST controller design method. The sampling interval is found by using the continuous equivalent of the discrete model (details are claimed proprietary). In an experiment the controller successfully controlled the temperature of the supply air after a heating coil by controlling the valve of the heating coil. In another experiment the controller successfully controlled the supply air duct pressure on a variable-air-volume (VAV) system. This design method is also a single-input single-output method.

In [4] (Zaheeruddin et al.) a single zone space heating system is represented by a seventh-order linear model. The heat is generated in a hot water tank (boiler) and is transferred to the zone. The temperature of the zone was controlled by controlling the fuel injection into the boiler or by controlling the hot water valve associated with the heating coil. An optimal proportional controller and an optimal PI (proportional-integral) controller were designed for a reduced order model of the heating system. Then, the PI controller was used to control the output of the full order model (sub-optimal control) and it was shown that the response was admissible. The sub-optimal controller was also implemented along with a state estimator to control the full-order model. Also, a multi-input controller (applying control on both the boiler fuel valve and the heating coil valve) was designed. All the controllers were designed by using a linearized model of the heating system. The

optimal controller design was based on minimizing a quadratic cost function. All the controllers were successful in regulating the temperature of the zone. It was shown that the response of the heating system is improved by using the multi-input controller.

In [5] (He et al.) a multivariable controller was designed for regulating a vapor compression cycle used in heating systems. An eleventh order and a reduced-order (fourth order) model described the system. The simulations results showed that the newly designed multivariable controller gave faster response to setpoint changes than the existing set of single-input single-output (PID) controllers. The multivariable controller design was based on optimizing a cost function. The purpose of the design was to get a good controlled system dynamic behavior around a certain steady state value of the variables. The authors predicted that the multivariable control would have wide applications in HVAC systems.

In [6] (Zaheeruddin et al.) partially decentralized controllers (which are multivariable controllers) are designed for a two-zone variable air volume heating system. Three controllers were designed for the three station (two zones and the air distribution system) for a VAV system. The controllers parameters were found by minimizing the cost function by using an optimization technique. The first controller, a two station decentralized controller, is designed to regulate the temperature of each zone by controlling the damper position of the zone. The third station output (the air flow rate in the distribution system) is not regulated. The second controller is a partially decentralized controller which regulates the two zones temperature and uses the output feedback signals of these stations to control the output of the third station, the air flow rate (by controlling the fan speed). The third controller is a fully decentralized controller which uses each station output to apply the control action on the corresponding input and regulating the output. All three controllers are shown to regulate the required outputs successfully. The cost function is computed from the signals of the nonlinear model and the parameter optimization directly

minimizes the cost function. Therefore, the linearized model of the VAV system is not required. The controllers are of constant gain type.

In [7] and [8], Curtis et al. have used neural network based control strategies in order to control (nonlinear) heating coils and the zones temperature of a two zone building heated by a heating ventilating and air conditioning (HVAC) system. The neural network based controller of the heating coil which is a predictive controller, controls the load of the coil. The controller of the overall system uses an energy saving strategy in order to minimize the energy (electrical energy) required for the control action. Experiments with a full scale real system proved the ability of the controllers in achieving the goals.

For controlling the load of a heating coil-valve combination a neural network was trained in order to learn the dynamic of this system. A number of such neural networks were used in a cascade form to predict the coil response at a sufficiently far future time to apply the control action. Another neural network, the controller, generated the control inputs. This neural network was trained, by using the back-propagation technique, such that the generated control inputs minimized the future error. The comparison of the neural network based controlled system response with a PID controlled system response showed that the neural network based controller responded faster and without any ripple.

The controller of the overall HVAC system also used a neural network for minimizing the energy consumed by the entire system due to the control action. The states of the HVAC system were measured over a range of operation. In each case the energy consumption of the HVAC system was calculated. Then, the neural network was trained to give the energy consumption as a function of the condition of the HVAC system.

The energy consumption minimization was carried out by a minimization technique which used the trained neural network for the required function evaluation. This minimization technique found a set of control variable values which controlled

the system with as little required energy as possible.

The performance of the system depends on the accuracy of prediction which becomes crucial when the system is first started, especially with unknown initial condition. Besides, if the predictor can not predict the process output accurately there is no guarantee that the control system will result in a stable closed-loop system. The neural networks in this system are trained off-line.

From the literature review we note that most HVAC controllers are designed for local loops by treating it as a single-input single-output system. Very few studies exist which use the multivariable design methods for HVAC systems. Some of these multivariable design methods use linear models. On the other hand the application of multivariable decentralized neural network controllers for HVAC applications have not been studied. The objective of this thesis is to explore the application of decentralized neural network controllers for HVAC systems. In particular the specified objectives of this thesis are:

1) To design a decentralized neural network controller for an MZSH system which requires regulation of three outputs using five inputs.

2) To compare the performance in terms of speed of response, disturbance rejection and cost minimization of the designed controller published [9].

3) To investigate different recall modes for practical implementation of the designed controller.

1.3 The Outline of this Thesis

This thesis consists of five chapters. Chapter one gives a general background about the subject and the goal of this thesis. In chapter two a background on multi-zone space heating systems is given. An analytical model of the multi-zone space heating system is described and open-loop characteristics of this heating system are studied. Some existing controllers which lead to the structure of the neural network based decentralized controller are explained next. Finally, the motivation for using the time-varying controller is described.

Chapter three gives the background on neural networks which is required for understanding the neural networks function in the neural network based decentralized controller. This chapter illustrates the nature and the operation of neural networks as well as a simple categorization of these networks. Also, the training of neural networks, the way it is used in this thesis, is presented.

Chapter four gives a detailed study of the neural network based decentralized controller. The structure of the controller and the neural network function as well as the training these networks in this controller are explained in detail. Numerical simulation results are given to prove the capability of the neural network based decentralized controller. Chapter five summarizes the conclusions of the thesis and gives some suggestions for future work.

Chapter 2

Background on Multizone Space Heating Systems

2.1 Introduction

The control of heating, ventilating and air conditioning (HVAC) systems associated with multizone buildings is of considerable interest because of the large amounts of energy used. In this type of problems, it is desired to obtain a controller in order to be able to control the temperature of a given zone of a building, independent of the temperature of the other zones, and independent of the outdoor climate of the building.

The usual practice of controller design for HVAC systems is based on feedback control using classical single-input/single-output control analysis, which means that the multivariable interaction of the control problem is generally ignored, therefore, severe problems of controller interaction may result [10]. The neural network based

decentralized controller design descussed in this thesis is based on modern control concepts and a parameter optimization method for finding an optimal controller [1]. Such a controller is designed for the whole system as an entity, thus different controller interactions are considered, and advantage is taken of decentralized control which simplifies the controller structure and its implementation.

This chapter is concerned with providing an understanding of the space heating system and some controller design methods so as to provide the framework for the neural network based decentralized control scheme. The available parameter optimization techniques as well as the one which is employed in this study along with required neural network concepts are presented in the next chapter.

The multizone space heating system is defined in the following section and the physical plant with its analytical model which are used in this study are presented. Then, robust controller design and decentralized control concepts are introduced. Three available regulator design methods are then presented. These design methods are

1) Linear robust servomechanism design,

2) Decentralized controller design,

3) Nonlinear decentralized servomechanism design.

The first design method uses the linearized model of the heating system in order to obtain a robust controller. The second method is essentially the same as the first one but in decentralized form. This method takes advantage of the structural properties of the plant (the heating system) in order to simplify the structure of the controller. The third design method looks into the problem from a nonlinear perspective, but the controller is still of the constant gain type. All these designs yield controllers which provide similar performance. They are included because the idea used in these design methods form the basis of the neural network based decentralized controller. At the end, the motivation for using a time varying gain controller is presented.

2.1.1 The Multizone Space Heating System

The heating systems that provide conditioned air to indoor spaces of buildings are called heating, ventilating and air conditioning (HVAC) systems. An example of such systems is shown in Figure 2.1. The system shown in Figure 2.1 consists of a boiler plus the corresponding heat conducting network and two rooms to which the heat energy is delivered. Because of modular structures of most buildings, like the one considered here, the corresponding heating system is called multizone space heating (MZSH) system. The system shown in Figure 2.1 is divided into three stations. Station 1 is the boiler and the heat conducting network and station 2 is zone 1 (or room 1) including heat delivering equipment and station 3 is zone 2 (or room 2) including its heat delivering equipment. The control action will be applied to the boiler and the heat delivering equipment of each zone. Although, only two zones are considered in this system the control problem to be solved can easily be extended to systems with more number of zones.

The boiler of the MZSH system (Figure 2.1) supplies warm water (SW) to the evaporative exchangers (E) of the heat pumps. Each zone has its own heat pump which works on the compression refrigeration cycle. It receives heat energy from the source water and increases its temperature to a higher level and then delivers the water to the condenser coil (C) of the heat pump. A circulating fan (F) and ductwork arrangement delivers the energy from condenser coil to the zone through the corresponding diffuser (D). The delivered energy compensates the heat loss due to cold ambient temperature such as that of a cold winter day.

The control problem of the MZSH system can be stated as follows. The output of each station must track some desired setpoint (reference input). In order to control the outputs of the control system, all inputs, i.e. $\nu_1, \cdots, \nu_5$, can be used to control each output, i.e. centralized control, or each station's input(s) can be used to control its output, decentralized control, i.e.,

1) The station 1 input $u_1(t) = \nu_3(t)$ is used to control its output $y_1(t)$. The

control is applied through C_3 in Figure 2.1.

2) The station 2 input $u_2(t) = [\nu_1(t)\ \nu_4(t)]'$ is used to control the station's output $y_2(t)$. The control is applied through C_1 and C_4 in Figure 2.1.

3) The station 3 input $u_3(t) = [\nu_2(t)\ \nu_5(t)]'$ is used to control the station's output $y_3(t)$. The control is applied through C_2 and C_5 in Figure 2.1.

Thus, the decentralized control of this heating system requires three independent controllers.

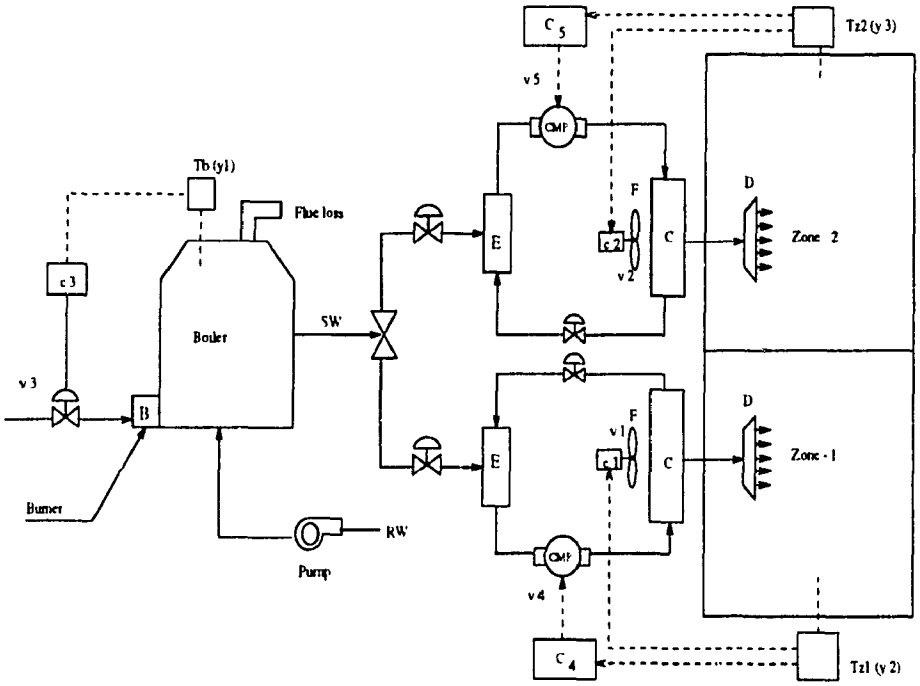

Figure 2.1: The diagram of Multizone Space Heating (MZSH) System.

CMP : Compressor

EV : Expansion valve

E: Evaporator

C : Condenser

D : Diffuser

F : Fan

C_i: Controllers ($i = 1, \cdots, 5$)

ν_i : Control inputs ($i = 1, \cdots, 5$)

y_j: outputs ($j = 1, 2, 3$)

SW : Supply water

RW : Return water

The following seventh order bilinear model described in [9] is used to simulate the MZSH system. Although, the differential equations which fully represent the behavior of this system are more complicated than what is considered here, this set of equations gives a good representation of the system.

$$\begin{aligned} c_b \dot{T}_b &= \nu_3 \nu_{3max} (1 - \alpha T_b / T_{bmax}) - m_b \, cp_w (T_b - T_{l1}) \\ &\quad - m_b \, cp_w (T_b - T_{l2}) - a_b (T_b - T_e) \end{aligned} \tag{2.1}$$

$$c_{l1} \dot{T}_{l1} = -\nu_4 \nu_{4max} (P_1 - 1) + m_b \, cp_w (T_b - T_{l1}) - a_{l1} (T_{l1} - T_e) \tag{2.2}$$

$$c_{h1} \dot{T}_{h1} = \nu_4 \nu_{4max} P_1 - \nu_1 \nu_{1max} \zeta (T_{h1} - T_{z1}) - a_{h1} (T_{h1} - T_e) \tag{2.3}$$

$$c_{z1} \dot{T}_{z1} = \nu_1 \nu_{1max} \zeta_1 (T_{h1} - T_{z1}) - a_{z1} (T_{z1} - T_p) - a_{z12} (T_{z1} - T_{z2}) \tag{2.4}$$

$$c_{l2} \dot{T}_{l2} = -\nu_5 \nu_{5max} (P_2 - 1) + m_b \, cp_w (T_b - T_{l2}) - a_{l2} (T_{l2} - T_e) \tag{2.5}$$

$$c_{h2} \dot{T}_{h2} = \nu_5 \nu_{5max} P_2 - \nu_2 \nu_{2max} \zeta (T_{h2} - T_{z2}) - a_{h2} (T_{h2} - T_e) \tag{2.6}$$

$$c_{z2} \dot{T}_{z2} = \nu_2 \nu_{2max} \zeta_2 (T_{h2} - T_{z2}) - a_{z2} (T_{z2} - T_p) - a_{z12} (T_{z1} - T_{z2}) \tag{2.7}$$

$$P_1 = 1 + (P_{1max} - 1)(1 - (T_{h1} - T_{l1}) / (\Delta T_{1max})) \tag{2.8}$$

$$P_2 = 1 + (P_{2max} - 1)(1 - (T_{h2} - T_{l2}) / (\triangle T_{2max})) \quad (2.9)$$

All the variables and parameters which are used in the above equations are defined in Table 2.1.

Table 2.1: Design parameters and the variables of the MZSH system [9]

Variable	Symbol	Magnitude, Unit
Boiler temperature	T_b	°C
Evaporatore temperature	T_{l1}, T_{l2}	°C
Comperassor temperature	T_{h1}, T_{h2}	°C
Zone temperature	T_{z1}, T_{z2}	°C
Out-door temperature	T_p	°C
Zone 1 Heat Loss Coefficient	a_{z1}	122.935 W/ °C
Zone 2 Heat Loss Coefficient	a_{z2}	138.32 W/ °C
Evaporator Heat Loss Coefficient	$a_{l1} = a_{l2}$	12.29 W/ °C
Condenser Heat Loss Coefficient	$a_{h1} = a_{h2}$	12.29 W/ °C
Boiler Heat Loss Coefficient	a_b	12.29 W/ °C
Interzone Heat Loss Coefficient	a_{z12}	12.29 W/ °C
Thermal Capacity of the Zones	$c_{z1} = c_{z2}$	374.48 kJ/ °C
Thermal Capacity of the Evaporators	$c_{l1} = c_{l2}$	167.44 kJ/ °C
Thermal Capacity of the Condensers	$c_{h1} = c_{h2}$	167.44 kJ/ °C
Thermal Capacity of the Boiler	c_b	594.55 kJ/ °C
Maximum Air Flow Rate	$\nu_{1max} = \nu_{2max}$	1.575 kg/s
Normalized Air Flow Rate	ν_1, ν_2	(dimensionless)
Burner Capacity	ν_{3max}	5.86 kJ/s
Normalized Burner Input	ν_3	(dimensionless)
Heat Pump Capacity	$\nu_{4max} = \nu_{5max}$	3.8 kJ/s
Normalized Heat Pump Input	ν_4, ν_5	(dimensionless)
Mass Flow Rate of Water	m_b	0.3151 kg/s
Specific Heat of Water	cp_w	4.186 kJ/kg °C
Heat Exchanger Coefficient	$\zeta = \zeta_1 = \zeta_2$	0.6 kJ/kg °C
Maximum Coefficient of Performance	$P_{1max} = P_{2max}$	3.5(dimensionless)
Maximum Temperature Differential	$\triangle T_{1max} = \triangle T_{2max}$	45 °C
Maximum Temperature of the Boiler	T_{bmax}	60 °C
Boiler Flue Loss Coefficient	α	0.1(dimensionless)

2.2 Open Loop Response of the Heating System

When a controller is designed for a specific purpose it is usually checked whether the controller is satisfying the associated requirements. But, in a general design, where no specific requirement can be stated, the controller performance is ususally compared with the performance of other controllers acting on the same system or it is compared with the open loop response of the system.

In this section two tests are presented which would let us have a perspective of the operation of the heating system. These tests are the drift of the initial state from the steady state response to a set of typical constant inputs and the open loop response of the heating system to a disturbance, which will be applied to the controlled system in Chapter 4, while the inputs are kept constant.

The response of the heating system to a constant input is a constant temperature if no disturbance is present. This constant input is called the nominal value of the input, which are usually used when dealing with the linearized model of a system. The steady state response of the heating system to this nominal input is the nominal state of the system. If the initial state of the heating system is drifted form its nominal value the response of the open loop heating system would reveal how fast or slow the system is. This response when compared with the response of the closed loop system is a good basis for understanding how fast the control system is responding to its test inputs.

Supposing that the inputs of the heating system is as follows,

$$[\ \nu_1\ \nu_2\ \nu_3\ \nu_4\ \nu_5] = [0.4\ 0.45\ 0.5\ 0.5\ 0.5] \tag{2.10}$$

the state values associated with this set of inputs are

$$[\ x_1\ x_2\ x_3\ x_4\ x_5] = \tag{2.11}$$

$$[T_b\ T_{l1}\ T_{h1}\ T_{z1}\ T_{l2}\ T_{h2}\ T_{z2}] = [27.26\ 25.58\ 29.84\ 21.96\ 25.51\ 27.94\ 20.66].$$

In order to check the open loop response of the heating system a two degree

increase in every state of the heating system is applied. Figures 2.2, 2.3, and 2.4 depict the outputs of the system starting from the initial states and generally reaching the nominal state values. These figures show that the system requires about two hours to have all station temperatures close to the nominal values.

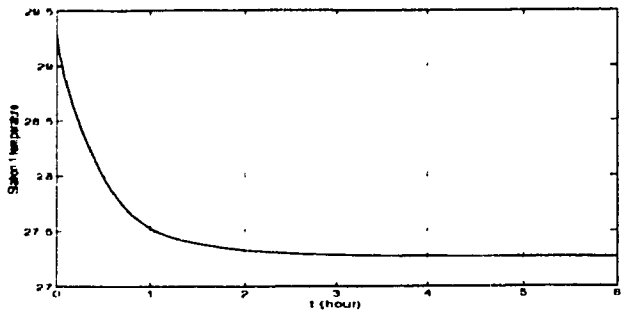

Figure 2.2: The temperature of the boiler (station 1) to a two degree increase of initial state in an open loop heating system.

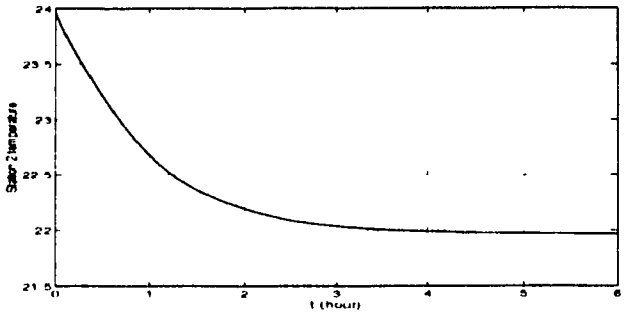

Figure 2.3: The temperature of zone 1 (station 2) to a two degree increase of initial state in an open loop heating system.

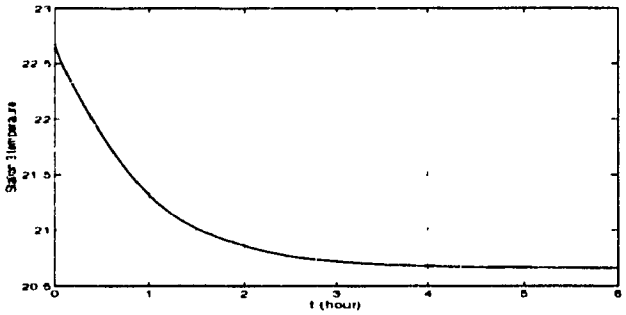

Figure 2.4: The temperature of zone 2 (station 3) to a two degree increase of initial state in an open loop heating system.

The open loop response of the heating system to a typical disturbance applied to such a system would be a basis for seeing how much the disturbance rejection characteristic of the heating system is improved when a controller is acting on it. The disturbance acting on the open loop system is shown in Figure 4.14 and the response of the heating system are depicted in Figures 2.5, 2.6, and 2.7.

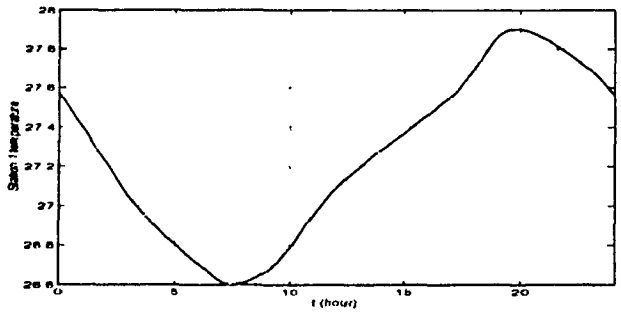

Figure 2.5: The temperature of the boiler (station 1) when the disturbance of Figure 4.14 is applied to the heating system.

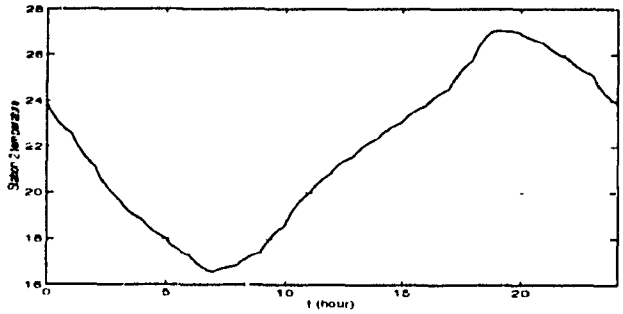

Figure 2.6: The temperature of zone 1 (station 2) when the disturbance of Figure 4.14 is applied to the heating system.

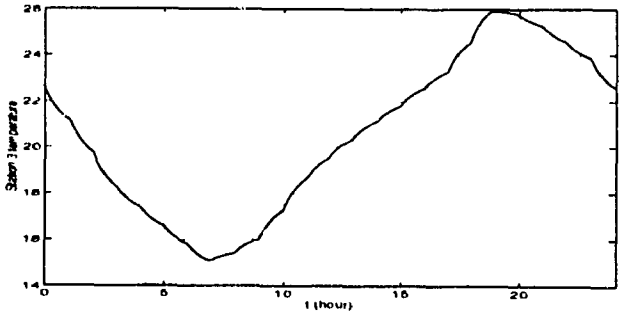

Figure 2.7: The temperature of zone 2 (station 3) when the disturbance of Figure 4.14 is applied to the heating system.

2.3 Constant Gain Controllers for the MZSH System

Constant gain controllers are of special interest in industry because of their simplicity of structure and compatibility with traditional control strategies which, despite the inefficiency in many cases, are still commonly used. Hence, much attention has been paid to design such controllers. This section gives some background of the structure and design of such controllers. Since the neural network based decentralized controller is a time-varying gain controller with the same structure as the constant gain controllers to be presented, such background is helpful in understanding the problem.

The robust servomechanism problem which yields the basic structure of the controller is briefly discussed first. Then the decentralized control problem which yields the complete structure of the controller is introduced. Next the linearized model of the MZSH system is obtained. This model is used for linear design purposes. Centralized and decentralized versions of the controller resulting from solving constrained linear servomechanism problems are also presented. The nonlinear decentralized servomechanism problem which has similarities to the neural network based controller design is discussed next. Finally, the motivation for using time varying controllers is given.

2.3.1 Robust Servomechanism Problem

The neural network controller is supposed to perform time varying control which is in essence a generalization of a gain scheduling type approach using piecewise constant gain controllers designed by linear approaches. The structure of the controller should

be the same as that of a linear controller. So it is desirable to study the linear controller structure here. A robust controller which can tolerate some gain margin, phase margin, and some nonlinearities of the plant is desirable in linear design methods. At the beginning of its training, a neural network output is not necessarily close to its optimal value, thus, the robustness of the controller structure assures the stability of the control system when this uncertainty in the behavior of the neural network may cause instability in the control system.

Any linear controller design should take into account the effect of perturbations in the model. This is because the linear equations

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{2.12}$$

$$y(t) = Cx(t) + Du(t) \tag{2.13}$$

which define the system model are usually obtained by linearization of a set of nonlinear equations about some operating point (x_0, z_0, u_0). Suppose the following nonlinear equations describe the system behavior more precisely:

$$\dot{x}(t) = f(x(t), z(t), u(t)) \tag{2.14}$$

$$\epsilon\dot{z}(t) = g(x(t), z(t), u(t)) \tag{2.15}$$

$$y(t) = h(x(t), z(t), u(t)) \,. \tag{2.16}$$

In equations 2.14-2.16, $z(t)$ corresponds to the high frequency modes of the system which are usually omitted in deriving the linear model. After linearization we have the following equations:

$$\delta\dot{x}(t) = A\delta x(t) + B\delta u(t) \tag{2.17}$$

$$\delta y(t) = C\delta x(t) + D\delta u(t) \tag{2.18}$$

with some condition on g [11].

In the above equations the exact values of A, B, C, or D may not be known or the operating point (x_0, z_0, u_0) may change. Thus, it is conceivable that the linear model in 2.17 and in 2.18 does not represent the exact behavior of the system about the chosen operating point. It is therefore important to take into account the effect of perturbations in the linear model when designing controllers based on such a model. The servomechanism problem which is presented in the following enables us to do this [10]-[13]. The resulting controller is robust in that it achieves the specifications for the servomechanism problem even in the presence of certain perturbations (non-destabilizing) in the linear model. A characteristic of the design is that the resulting controller contains a certain duplicated model of its environment, i.e. of the disturbances and the reference inputs acting on the system. This characteristic of the design is known as the *Internal Model Principle* [14].

Consider a linear time-invariant system described by

$$\dot{x}(t) = Ax(t) + Bu(t) + Ed(t) \tag{2.19}$$

$$y(t) = Cx(t) + Du(t) + Fd(t) \tag{2.20}$$

$$e(t) = y(t) - y_r(t) \tag{2.21}$$

where $x(t) \in R^n$ is the state vector, $u(t) \in R^m$ is the input vector, $y(t) \in R^p$ is the output vector, $d(t) \in R^l$ is the disturbance vector which may or may not be measurable, $y_r(t) \in R^p$ is the reference signal vector and $e(t) \in R^p$ is the output error vector. The disturbance vector $d(t)$ is assumed to be generated by a system

represented by the following equations:

$$d(t) = C_d z_d(t) \tag{2.22}$$

$$\dot{z}_d(t) = A_d z_d(t) \tag{2.23}$$

which is observable. Likewise, the reference signal vector $y_r(t)$ is of the form

$$y_r(t) = C_r z_r(t) \tag{2.24}$$

$$\dot{z}_r(t) = A_r z_r(t) \tag{2.25}$$

which is observable. A controller for the system of equations 2.19, 2.20 is said to be robust if the given system can be controlled by it in the desired manner, in spite of the allowable disturbances and changes in the system parameters. Thus, the control problem can be stated as follows. The initial condition of z_d in 2.23 may be unknown while those of z_r in 2.25 are assumed to be known.

It is required to construct a controller for the system 2.19 - 2.21, using the available measurements $y(t)$ so that the resulting controlled system is stable and the steady state error is zero (i.e. asymptotic regulation takes place) for all disturbances $d(t)$ and reference signals $y_r(t)$ satisfying equations 2.21-2.23 and 2.24- 2.25 respectively, independent of any allowable perturbations in the system parameters.

The only restriction on the perturbations in the system parameters is that they should not make the overall closed-loop system unstable.

If (i) (A,B) is a stabilizable pair and (ii) (C,A) is a detectable pair and (iii) the number of inputs is greater than or equal to the number of outputs, i.e. $m \geq p$ and (iv) and

$$rank \begin{bmatrix} A - \lambda_i I & B \\ C & D \end{bmatrix} = n + p, \qquad i = 1, 2, \cdots, q \tag{2.26}$$

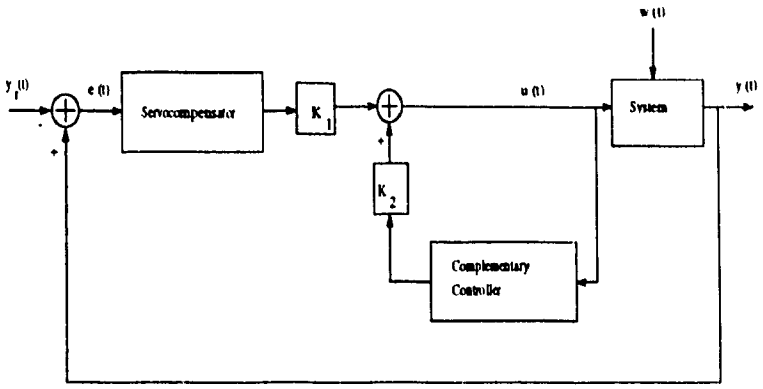

Figure 2.8: A servo compensator.

(where λ_i is an eigenvalue of A_d or A_r) a robust controller can be designed by using $y(t)$, such that the closed-loop system is stable and asymptotic regulation takes place for all $d(t)$ and $y_r(t)$ defined above regardless of any (allowable) perturbations in the system parameters. Such a controller consists of two distinct parts a servocompensator, which is completely determined by the disturbances and reference signals, and a stabilizing controller which can be implemented via an observer providing an estimate of plant states. The structure of the robust control system is given in Figure 2.8, [14].

For the HVAC system at hand it was shown ([9]) that the plant is stabilizable using output feedback, thus, output feedback can be used instead of state feedback. The structure of the servocompensator for the system is as follows [10].

$$\dot{\xi}_i(t) = \Omega_i^* \xi_i(t) + \Theta_i^* e_i(t), \quad i = 1, \cdots, p \tag{2.27}$$

The structure of this servocompensator is the same as that of the disturbance and the reference input related to it. For constant reference inputs and disturbances which have no dynamics the Ω_i^* would be zero and Θ_i^* would be equal to an identity matrix. Thus, equation 2.27 defines an integrator and the controller would be a P.I. controller [9].

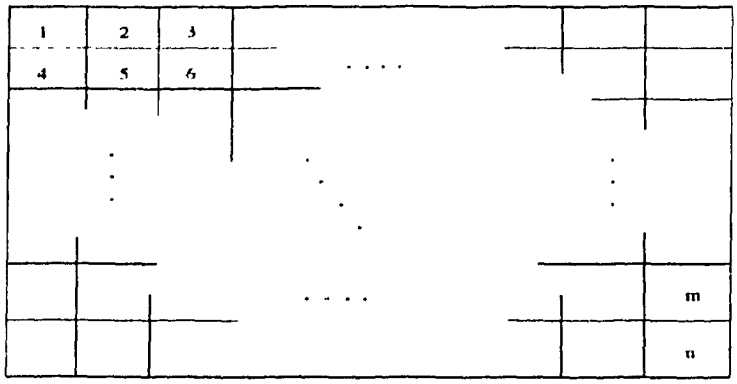

Figure 2.9: A simple representation of a multizone system. The temperature of zone 6 or zone n is not likely to affect the temperature of zone 1.

2.3.2 Decentralized Control

Because of the modular structure of most buildings the HVAC system which delivers energy from a hot water source to building rooms is called a multi-zone space heating (MZSH) system. Having isolated zones, such systems are usually controlled by using totally independent control loops each acting on one zone.

In practice different zones of a building are not completely isolated. Because of heat transfer taking place due to conduction of walls, roofs, and ceilings, and the convection between surfaces and air and by radiation between surfaces, the general problem of multi-zone temperature control leads to a multivariable control problem. But, if the heat transferred from one room to another is negligible, compared with the heat delivered by the heating ventilating and air conditioning system, the interaction of the zones is negligible and a set of single-input single-output controllers can provide the satisfactory temperature control.

Also, because of several temperature control zones in a building, which may not be adjacent to each other, it makes sense to restrict the structure of the control system so that local feedback loops are used as much as possible (Figure 2.9). This simplifies the structure of the controller and results in what is called a decentralized controller [15].

In a real system the heat exchange of some zones may be considered while the heat exchange of others may be neglected [15]. The controller of a zone would use the signals available in zones which are expected to considerably affect this zone. Therefore, the decentralized structure of the controller may vary depending on the requirements of the building. However, the interaction between zones is considered in the design of such controllers by treating the system as a multivariable system. The controller gains obtained from a decentralized controller design are not necessarily the same as those obtained from a centralized controller design.

A decentralized controller can not perform better than a centralized controller. This is because the class of decentralized controllers is a subset of that of decentralized controllers.

Referring to the linear model 2.19 - 2.21, the decentralized system is given by

$$\dot{x}(t) = Ax(t) + \sum_{i=1}^{N} B_i u_i(t) + Ed(t) \tag{2.28}$$

$$y_i(t) = C_i x(t) \qquad i = 1, \cdots, N \tag{2.29}$$

$$\epsilon_i(t) = y_i(t) - y_{r_i}(t) \qquad i = 1, \cdots, N \tag{2.30}$$

where the system is assumed to have N stations and $u_i(t) \in R^{m_i}$, $y_i(t) \in R^{l_i}$, and $y_{r_i}(t) \in R^{l_i}$ ($\sum m_i = m$ and $\sum l_i = l$).

2.3.3 Linearized Model of the Heating System

Linear design methods require a linearized model of the plant. The setpoint which is considered here is about the midrange of inputs, allowing the regulator to operate over small changes in setpoints without losing the validity of the linearization. The input values are

$$[\nu_1(t)\ \nu_2(t)\ \nu_3(t)\ \nu_4(t)\ \nu_5(t)]' \ = \ [0.4\ 0.45\ 0.5\ 0.5\ 0.5]' \qquad (2.31)$$

and the constant disturbance which is needed for finding the linearized model is

$$d(t) \ = \ [20\ -2]'. \quad (^oC) \qquad (2.32)$$

The above conditions yield the following state of the heating system at which the model is linearized.

$$[x_1(t)\ x_2(t)\ x_3(t)\ x_4(t)\ x_5(t)\ x_6(t)\ x_7(t)]' = \qquad (2.33)$$
$$[T_b(t)\ T_{l1}(t)\ T_{h1}(t)\ T_{z1}(t)\ T_{l2}(t)\ T_{h2}(t)\ T_{z2}(t)]' =$$
$$[27.2661\ 25.5844\ 29.8430\ 21.9668\ 25.5120\ 27.9441\ 20.6617]' \quad (^oC)$$

For the following linear model of the plant, we can find the matrices A, B, and C and E by using MATLAB.

$$\triangle \dot{x}(t) \ = \ A\ \triangle x(t) + B\ \triangle u(t) + E\ \triangle d(t) \qquad (2.34)$$

$$\triangle y(t) \ = \ C\ \triangle x(t) \qquad (2.35)$$

The matrices are given in Appendix A. This model is valid for

$$x_{i,nom} - 1.5 \leq x_i(t) \leq x_{i,nom} + 1.5\ . \qquad (2.36)$$

Where $x_{i,nom}$ denotes the nominal state of the system. The decentralized linearized model of the plant is given by

$$\triangle \dot{x}(t) = A\ \triangle x(t) + \sum_{i=1}^{3} B_i\ \triangle u_i(t) + E\ \triangle d(t) \qquad (2.37)$$

$$\triangle y_i(t) \ = \ C_i\ \triangle x(t) \qquad (2.38)$$

where the matrices are given in Appendix A and Δ represents the current value of a variable minus its nominal value.

The inputs and outputs of the decentralized system are defined as follows and remains the same in different approaches to the solution of the problem.

$$u_1(t) = \nu_3(t), \quad u_2(t) = \begin{bmatrix} \nu_1(t) \\ \nu_4(t) \end{bmatrix}, \quad u_3(t) = \begin{bmatrix} \nu_2(t) \\ \nu_5(t) \end{bmatrix}, \tag{2.39}$$

and

$$y_1(t) = x_1(t), \quad y_2(t) = x_4(t), \quad y_3(t) = x_7(t) \tag{2.40}$$

2.3.4 Linear Robust Servomechanism Problem

In order to find a robust controller for the heating system, without considering its decentralized structure, the following centralized control problem is solved

$$\begin{aligned} Min\ J(K) &= trace\{\int_0^t [e(t)'Qe(t) + u(t)'Ru(t)]\,dt\} \\ &= trace\{P\} \end{aligned} \tag{2.41}$$

subject to

$$\Delta \dot{\bar{x}}(t) = \bar{A}\ \Delta \bar{x}(t) + \bar{B}\,\Delta u(t) + E\ \Delta d(t) + F\,\Delta y_r(t) \tag{2.42}$$

$$\Delta \bar{y}(t) = \bar{C}\,\Delta \bar{x}(t) = \begin{bmatrix} \Delta y(t) \\ \Delta \xi(t) \end{bmatrix} \tag{2.43}$$

$$0 \le \nu_k(t) \le 1, \quad k = 1, \cdots, 5 \tag{2.44}$$

$$Rc(\lambda_j^c) < 0 \tag{2.45}$$

where K is the controller gain matrix. Q and R are positive definite matrices. $\Delta\xi \in R^{pq\times 1}$ is the output of the servocompensator, i.e.,

$$\Delta\dot{\xi}(t) = \Omega\,\Delta\,\xi(t) + \Theta\,\Delta\,e(t)\ . \tag{2.46}$$

$$\Delta\,e(t) = \Delta y(t) - \Delta y_r(t) \tag{2.47}$$

The matrices $\bar{A}$, $\bar{B}$, $\bar{C}$, $\bar{E}$, and $\bar{F}$ are given by

$$\bar{A} = \begin{bmatrix} A & 0 \\ \Theta C' & \Omega \end{bmatrix}, \bar{B} = \begin{bmatrix} B \\ 0 \end{bmatrix}, \bar{E} = \begin{bmatrix} E \\ 0 \end{bmatrix}, \bar{F} = \begin{bmatrix} 0 \\ -\Theta \end{bmatrix}, \bar{C} = \begin{bmatrix} C & 0 \\ 0 & I_{pq} \end{bmatrix} \tag{2.48}$$

and Ω and Θ are $0_{3\times 3}$ and $I_{3\times 3}$. In the above equations q is the order of the least common multiple of the minimal polynomials of the disturbances and the reference inputs defined by equations 2.22-2.25 [16]. It is simply equal to one for all disturbances and reference inputs assumed to be constant. p is the total number of the outputs of the MZSH system which is three here. Condition 2.44 ensures that the solution of the problem is feasible (does not violate the limits of the control inputs) and 2.45 ensures that the control system remains stable while the controller gains are being adjusted. The matrix $\bar{K}$ is defined by:

$$(\bar{A} + \bar{B}\bar{K}\bar{C})' + \bar{P}(\bar{A} + \bar{B}\bar{K}\bar{C}) = (Q + \bar{C}'\bar{K}'R\bar{K}\bar{C}) \tag{2.49}$$

$$\Delta\,u(t) = \bar{K}\begin{bmatrix} \Delta e(t) \\ \Delta\xi(t) \end{bmatrix}, \tag{2.50}$$

where $\bar{K} = \left[\bar{K}^{(1)}\ \bar{K}^{(2)}\right]$. This constraint optimal control problem was solved in [9] using a parameter optimization program. The method of solution requires starting from some stabilizing controller gain and obtaining P by solving 2.49 and applying a sequential search type parameter optimization technique to find $\bar{K}$ which minimizes the value of P while satisfying the constraint in 2.41-2.45

The controller obtained in 2.49 by this design method was

$$\bar{K} = \begin{bmatrix} 0.26821 & -0.00796 & 0.02346 & 0.66410 & 0.08757 & 0.06621 \\ 0.04060 & 0.31637 & 0.02575 & 0.05495 & 0.78372 & 0.04820 \\ 0.03671 & 0.31706 & 0.02381 & 0.04882 & 0.80676 & 0.06689 \\ 0.05082 & 0.03793 & 0.18882 & 0.04647 & 0.05178 & 1.08092 \\ 0.01113 & 0.03099 & 0.25037 & 0.03194 & 0.05277 & 0.77449 \end{bmatrix}. \quad (2.51)$$

2.3.5 Decentralized Controller Problem

It was shown in [15] that the temperature of a zone in a linear system can be controlled by using output feedback and using signals available from the immediate neighboring zones (but not other zones). An output feedback decentralized controller using only each zone output can be designed which gives performance close to the performance of a centralized controller for a plant the same as the one in this study [9].

The augmented decentralized control system using the linearized model of the plant is represented by the following equations.

$$\Delta \dot{\hat{x}}(t) = \hat{A} \Delta \hat{x}(t) + \sum_{i=1}^{3} \hat{B}_i \Delta u_i(t) + \hat{E} \Delta d(t) + \hat{F} \Delta y_r(t) \quad (2.52)$$

$$\Delta \hat{y}_i(t) = \begin{bmatrix} \Delta y_i(t) \\ \Delta \xi(t) \end{bmatrix} = \hat{C}_i \Delta \hat{x}(t), \qquad i = 1, 2, 3 \quad (2.53)$$

$$\Delta \dot{\xi}_i(t) = \Omega_i^* \Delta \xi_i(t) + \Theta_i^* \Delta e_i(t) \qquad i = 1, 2, 3 \quad (2.54)$$

and

$$\triangle e_i(t) = \triangle y_{r_i}(t) - \triangle y_i(t) \tag{2.55}$$

where

$$\hat{x}(t) = \begin{bmatrix} \triangle x(t) \\ \triangle \xi(t) \end{bmatrix}, \triangle \hat{y}_i(t) = \begin{bmatrix} \triangle y_i(t) \\ \triangle \xi(t) \end{bmatrix} \tag{2.56}$$

In equation 2.54, Ω_i^* and Θ_i^* are zero and one respectively which are defined by the constant reference signal and the constant disturbance acting on the system. This leads to the following servocompensator

$$\triangle \dot{\xi}(t) = \Omega^* \triangle \xi(t) + \Theta^* \triangle e(t) \tag{2.57}$$

with $\Omega^* = 0_{3\times 3}$ and $\Theta^* = I_3$.

$$\hat{A} = \begin{bmatrix} A & 0 \\ \Theta^* C & \Omega^* \end{bmatrix}, \hat{B}_i = \begin{bmatrix} B_i \\ 0 \end{bmatrix}, \hat{E} = \begin{bmatrix} E \\ 0 \end{bmatrix}, \hat{F} = \begin{bmatrix} 0 \\ -\Theta^* \end{bmatrix}, \hat{C}_i = \begin{bmatrix} C & 0 \\ 0 & I_p \end{bmatrix} \tag{2.58}$$

$\triangle$ before a variable represents the current value of the variable minus the nominal value of the variable. The appropriate matrices are given in Appendix A. Note that all the matrices can be obtained from the centralized linear model of the heating system.

Since in decentralized control problem there is no control action between stations the controller gain matrix K is block-diagonal, i.e.,

$$K = block\ diag(K_1, K_2, K_3) \quad K_1 \in R^{1\times 2},\ K_2 \text{ and } K_3 \in R^{2\times 2} \tag{2.59}$$

The measure of a decentralized control structure gives rise to what is called the problem of decentralized fixed modes (DFMs). Decentralized fixed modes concept is a generalization of the concept of uncontrollable and/or unobservable modes of centralized linear systems. A DFM is that mode of a system which is not affected by

all possible controller gains of the chosen decentralized controller structure. It only makes sense to use a particular decentralized structure if all the corresponding DFMs are asymptotically stable. For the MZSH system, there are no unstable DFMs, and the number of inputs (N_i) for each station is greater than or equal to the number of outputs (l_i). Then for this system the robust decentralized control system is stable if the triple (A, B, C) has no transmission zeros at the origin $s = 0$ (constant references and disturbances). Using MATLAB it was found that the system has no finite transmision zeros and the poles are at -1.2183, -45.1572, -1.7729,-1.2410, -28.6250, -0.2642, -0.2642. The latter implies that there are no unstable DFMs.

Therefore, this decentralized servomechanism problem has a solution. The problem was again formulated as a constrained optimal control problem in [9] with a decentralized control gain defined by

$$\Delta u_i(t) = \tilde{K}_i \begin{bmatrix} \Delta e_i(t) \\ \Delta \xi_i(t) \end{bmatrix}, \quad i = 1,2,3 \tag{2.60}$$

$$\tilde{K}_i = \left[\tilde{K}_i^{(1)} \;\; \tilde{K}_i^{(2)}\right]. \tag{2.61}$$

To have a gain matrix with the same form as the one found by the centralized approach, we may rearange the above matrix as:

$$K = \left[K^{(1)} \;\; K^{(2)}\right], \tag{2.62}$$

$$K^{(1)} = block\ diag\{\tilde{K}_1^{(1)} \;\; \tilde{K}_2^{(1)} \;\; \tilde{K}_3^{(1)}\},\ K^{(2)} = block\ diag\{\tilde{K}_1^{(2)} \;\; \tilde{K}_2^{(2)} \;\; \tilde{K}_3^{(2)}\}$$

and

$$\Delta u(t) = K \begin{bmatrix} \Delta e(t) \\ \Delta \xi(t) \end{bmatrix}, \Delta u(t) = \begin{bmatrix} \Delta u_1(t) \\ \Delta u_2(t) \\ \Delta u_3(t) \end{bmatrix}, e(t) = \begin{bmatrix} \Delta e_1(t) \\ \Delta e_2(t) \\ \Delta e_3(t) \end{bmatrix}, \xi(t) = \begin{bmatrix} \Delta \xi_1(t) \\ \Delta \xi_2(t) \\ \Delta \xi_3(t) \end{bmatrix} \tag{2.63}$$

A similar cost function as that of the centralized case was used. Thus, the minimization problem is as follows.

$$Min\ J(K) \;\; = \;\; \int_0^t \left[\Delta e(t)' Q\, \Delta e(t) + \Delta u(t)' R\, \Delta u(t)\right] dt\ . \tag{2.64}$$

By using the approach given in the previous section, the following decentralized feedback gain matrix was obtained in [9].

$$K = \begin{bmatrix} 0.3230 & 0 & 0 & 0.7759 & 0 & 0 \\ 0 & 0.3420 & 0 & 0 & 0.7205 & 0 \\ 0 & 0.3477 & 0 & 0 & 0.7868 & 0 \\ 0 & 0 & 0.1865 & 0 & 0 & 1.0752 \\ 0 & 0 & 0.2697 & 0 & 0 & 0.7588 \end{bmatrix}. \tag{2.65}$$

The comparison of this set of feedback gains and the one shown in 2.51 reveals the difference between the control action for the centralized and decentralized cases.

2.3.6 Nonlinear Decentralized Servomechanism Problem

Here a similar approach was used as in the preceding section except that the linearized model was replaced by the actual bilinear model of plant:

$$\dot{x}(t) = \tilde{A}x(t) + \sum_{i=1}^{3} \tilde{B}_i(x)\, u_i(t) + \tilde{E}d(t) \tag{2.66}$$

$$y_i(t) = \tilde{C}_i x(t) \quad i = 1, 2, 3 \tag{2.67}$$

$$e_i(t) = y_i(t) - y_{r_i}(t) \quad i = 1, 2, 3 \tag{2.68}$$

where

$$x(t) = [T_b(t)\ T_{l1}(t)\ T_{h1}(t)\ T_{z1}(t)\ T_{l2}(t)\ T_{h2}(t)\ T_{z2}(t)]', \tag{2.69}$$

$$d(t) = [T_e(t) T_p(t)]'. \tag{2.70}$$

The servocompensator can be defined as before, i.e.,

$$\dot{\xi}_i(t) = \tilde{\Omega}_i \xi_i(t) + \tilde{\Theta}_i e_i(t), \quad i = 1,2,3 \tag{2.71}$$

with $\tilde{\Omega}_i = 0$ and $\tilde{\Theta}_i = 1$.

$$u_i(t) = \tilde{K}_i \begin{bmatrix} e_i(t) \\ \xi_i(t) \end{bmatrix}, \quad i = 1,2,3 \tag{2.72}$$

where $\tilde{K}_i = \left[\tilde{K}_i^{(1)} \ \tilde{K}_i^{(2)}\right]$. The structure of the controller in this design is the same as what introduced in the previous section.

Supposing that a 24 hour outdoor temperature forecast, T_p, is available the following optimization problem can be solved which results in the controller gain matrices of the decentralized structure.

$$Min \ \tilde{J}(\tilde{K}_1, \tilde{K}_2, \tilde{K}_3) = \int_0^{24} [e(t)'Qe(t) + u'(t)Ru(t)]\, dt \tag{2.73}$$

subject to

$$0 \le \nu_i(t) \le 1 \qquad i = 1, \cdots, 5 \tag{2.74}$$

Q and R are the same as those of previous designs.

2.4 Motivation for Time Varying Controller

Considering the entire control operation required in an MZSH control problem, we need a regulator which can perform over a larger range than what was assumed for the controllers in the previous section. We may be able to use one controller, over a wide range of operation, provided it is robust over the range. However, there will still be significant degradation for larger ranges especially at their limits. So using a robust constant controller is not a good solution to the problem. Also, since the regulator is not optimal for other operating points, we may get a response which converges to the desired setpoint values, (assuming that the system is stable), but the transient response of the system may not be fast. This fact is of particular relevance for controllers designed using linearized models.

Since the plant is nonlinear, a better controller from a linear design point of view is a time varying controller. Suppose for designing a controller for the whole operating range we divide the range into several small sub-ranges and design a controller for each sub-range. We would then have several controllers, each designed for a particular point in a sub-range, and optimal at the operating point of the sub-range. The controller's gain matrix is given by

$$K = K_{(i)} \quad (X \in Range\ i) \quad i = 1, \cdots, M \tag{2.75}$$

where X denotes the state vector of the control system. As the operation switches from one sub-range to another, the controller switches from one value to the next. This is essentially the principle of gain scheduling that is frequently used for controlling nonlinear systems. If we increase M we get more controllers each optimal over a more restricted sub-range (and presumably performing better over that sub-range). In the limit when M approaches infinity we would have a function, say $K(X)$, which gives us an optimal controller gain depending the state of the plant. The controller which is the goal of this study provides a gain matrix which is a function of the states of the system.

Since output feedback and PI control have been shown to be able to control the MZSH system, it is reasonable to suppose that outputs of the plant and the variables in the PI controller can represent the system condition adequately.

Chapter 3

Background on Neural Networks

3.1 Introduction

Neural networks are parallel distributed computing networks which consist of nodes (or neurons) and their interconnections (or weights). Neural networks, unlike usual man made machines are not designed specifically for what they are employed for. They are designed heuristically i.e. a designer defines neural network inputs, its architecture, proper neurons, initial weights, and training mode and then lets the network learn its task. Training a neural network means changing its internal parameters according to some training algorithm.

Neural networks can have a variety of different architectures. Among them multi-layer neural networks are widely used for control purposes because of the use of the backpropagation training scheme ([17], [18]) and relatively little computation required when neural networks are employed in recall mode.

Three important characteristics of neural networks will potentially help to understand their function in this project. These characteristics are (i) learning a function of some variables, (ii) minimizing some energy function, and (iii) generalizing the already learned data. These characteristics are specially desirable for finding optimal controllers for nonlinear systems. Since controller parameters are most likely functions of some of the control system states the first characteristic attracts attention. Optimality of the controller requires minimization of some cost function which can be done by using a neural network. Training a neural network for all possible conditions of a system is impossible. So generalization of the learned data is a necessity.

This chapter introduces the architecture and the role of the neural networks employed in a neural network based decentralized controller. In order to give some background about neural networks, the main characteristics of the networks are presented. Training a single neuron and using backpropagation training, a widely used training algorithm, will be explained next. Learning a function and doing optimization are given in subsequent sections. Then, function learning capability of neural networks is discussed. Finally, performing optimization by using neural networks is considered.

3.2 Neural Networks

Neural networks are parallel distributed computing networks which accomplish their tasks with parallel and redundant computations rather than doing them sequentially and in order. They are represented by nodes (neurons) and their connections which are represented by arrows going out from one node to another node. The direction of these arrows represent the direction of information flow i.e. showing that the output of one node is affecting the input of another. Figure 3.1 depicts a simple neural network.

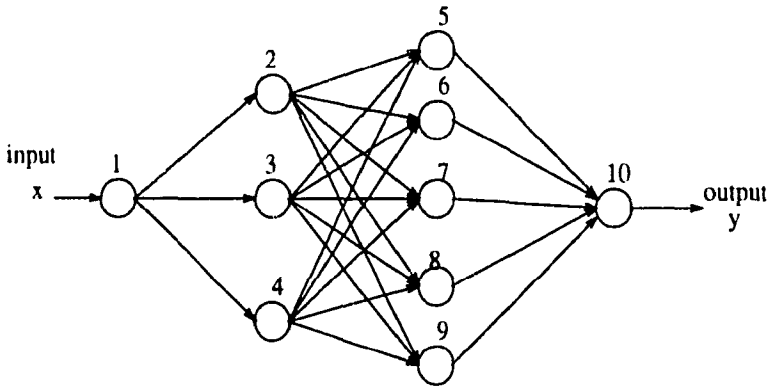

Figure 3.1: A node arrow representation of a neural network.

Node 1 is the input terminal which is usually considered as a buffer to the neural network. Nodes 2 to 9 are neurons of the network. As one can see the information passes from node 1 to nodes 2-4 and from there to nodes 5-9 and so on. Node 10 is the output node which is also a neuron.

The characteristic of neurons and the way the neurons are connected, which is called the topology of the network, constitute the neural network input/output mapping. In Figure 3.1 suppose that all the nodes are of symmetric type then it is possible for the network to be symmetric i.e.

$$x \rightarrow y \tag{3.1}$$

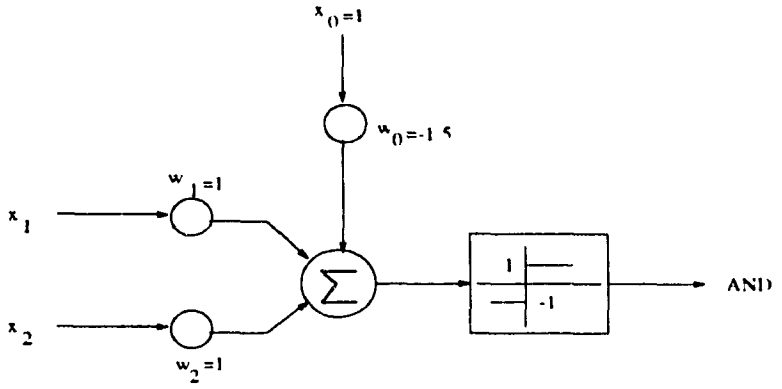

Figure 3.2: A neural network model of logic AND.

$$-x \rightarrow -y$$

which in not the case when the neurons are not symmetric.

In transferring the output of one neuron to the input of another, the connection also multiplies the output by some scalar which is called the weight of this connection. These weights in a neural network which are usually put in groups and represented by some matrices can also modify, or sometimes totally change, the characteristics of the neural network. For example Figure 3.2 shows an AND logic operator. But, by changing only w_0 from -1.5 to zero the operator becomes an OR [19]. In more complex networks the circles which represent the scaling are omitted and each weight (w_i) is placed over the proper arrow.

When a particular neural network is chosen by the designer the weights, which are usually considered as internal parameters of the network are found, adjusted, and fine tuned in order to let the neural network perform its function. Since the parameter, or weight adjustment allows the neural network to perform better it is said that the network is *learning*.

The weight adjustment is accomplished either by the interaction of nodes due to the circulation of information in the neural network or by a training algorithm which finds weight changes from some rule. These two forms of weight adjustments are called unsupervised and supervised learning respectively. If supervised learning

is what is employed in a system, the training algorithm can be represented by a supervisory block which adjusts the neural network, Figure 3.3. This supervisory

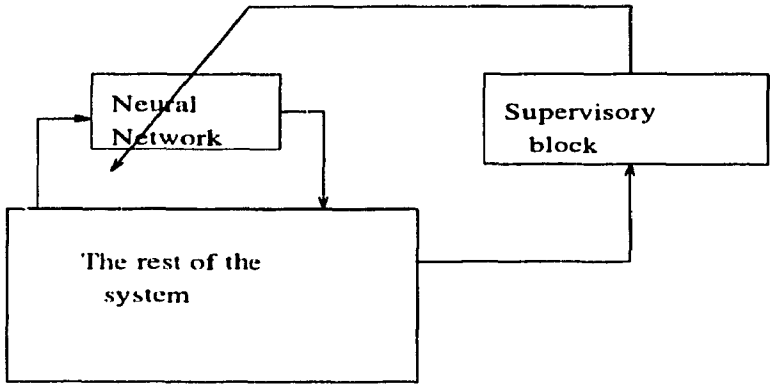

Figure 3.3: Supervised learning.

block may be removed from the block diagram for the sake of simplicity. Because of the flexibility of their structure, neural networks are difficult to classify. Some attempts have been done in the past in order to compare different neural networks [20]. The following presents a simple classification [17].

1. Topology
2. Architecture
3. Neuron model
4. Training algorithm
5. Operation schedule

The topology of a neural network refers to how the neurons are connected in the network. For instance the network shown in Figure 3.1 is a feedforward network where neurons closer to the output cannot affect the neurons which are farther from the output. There are networks which employ such connections. The internal parameters, or weights, of a neural network can be considered as a part of the topology of the network because a disconnection can be represented by a connection with a weight of zero.

The architecture of a neural network denotes how the inputs of the network are defined and whether the network has any feedback from its output(s) to its input(s).

Neurons which are depicted as nodes in Figure 3.1 are the most important parts of the structure. They, in essence, create the flexibility which a neural network is known for.

An operation schedule determines the order in which different parts of a neural network should be activated. This schedule is important for stability of some neural networks.

3.2.1 Topology

The topology of a neural network is the pattern of connections between inputs, neurons, and outputs. It specifies the neural network and the information stored in it, i.e., it includes what the system knows and determines how it responds to any arbitrary input.

A topology which covers all other topologies is the fully connected topology, e.g., bidirectional associative memory. This topology requires that every node's output be connected to an input of all the neurons inside the neural network including an input of its own [21]. Thus other topologies can be considered as fully connected topologies with some connections equal to zero. Some of the commonly used topologies are multilayer neural networks, bidirectional associative memories, and ARTs [22].

A very common topology is the multilayer feedforward, or hierarchical topology neural network (Figure 3.1). This topology orders neurons into layers with neurons whose outputs are connected to the next layer neurons inputs and their inputs connected to the previous layer neurons outputs. So, the information flows in one

direction.

Because of the importance of multilayer networks in applications there is an interest in finding the number of layers and neurons required for a specific job. Although a clear set of rules for them has not yet been obtained some general characteristics can be stated. If all the neurons of a network have nonlinear activation functions, increasing the number of layers generally enables the network to learn more complex patterns [24]. Two layer networks, i.e. one hidden layer network, was shown to be able to approximate a wide range of nonlinear functions [24]. But it is suggested that a three layer network, i.e. with two hidden layers, may have some advantages [23].

The number of neurons required for a task is also unknown. Increasing the network layers may help reducing the total number of neurons required for the same task. It should be noted that increasing the number of layers and neurons imposes heavier computation requirements for practical problems, and may not even improve performance.

All neurons in a layer are usually chosen to be of the same type but different layers may have different types of neurons.

3.2.2 Architecture

A simple categorization of neural network architectures consists of feedforward, external feedback, internal feedback, and unsupervised.

A Feedforward network responds to its input at every instant. An internal feedback neural network performs like a feedforward network but its output affects the input. Thus, the response to the stimulus, which is a combination of the external stimulus and the feedback, is defined no matter what the previous external

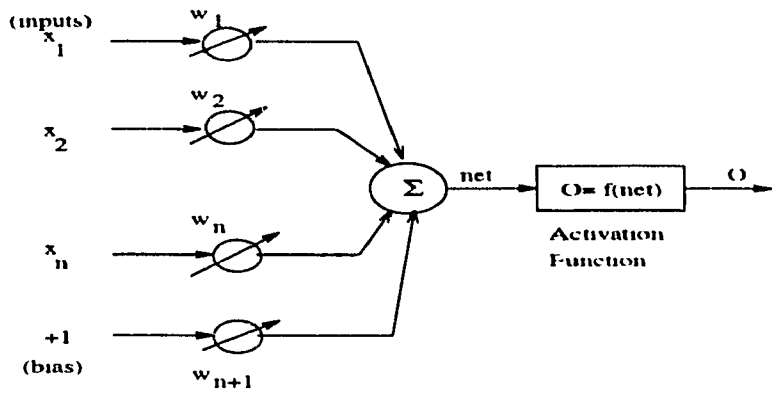

Figure 3.4: The model of a neuron

stimuli have been. An external feedback network sees both the external stimulus and its own response to the previous stimulus. Thus, its response depends on the current stimulus, the previous stimuli, and the order in which they were presented to the network. The unsupervised network, for example, the winner-take-all architecture can-not be assigned a desired output and the network responds to some characteristic of the stimulus which it can learn. Feedforward, external feedback, and internal feedback architectures can undergo training schemes but unsupervised networks cannot.

3.2.3 Neuron Model

A neuron is the basic building block of a neural network. Regardless of different architectures which can be employed for a purpose a neuron is the essential block which remains unchanged. An artificial neuron is a model of a biological neuron and can be represented as shown in Figure 3.4.

where $x_1, \cdots, x_n$ are the inputs to the neuron. Each input is scaled by a weight w_i and the scaled inputs are added and shifted (by the *bias* w_{n+1}) to provide

the input to the activation function. i.e.

$$\begin{aligned} net &= W'x + w_{n+1}bias \\ y &= f(net) \end{aligned} \tag{3.2}$$

where

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \quad W = \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix} \tag{3.3}$$

A variety of activation functions are conceivable. Some of them are

1) Hard-limiting (or binary perceptron)

$$f(net) = \begin{cases} 1 & net > 0 \\ 0 & net \leq 0 \end{cases} \tag{3.4}$$

2) Soft-limiting (or continuous perceptron or tan-sigmoid)

$$f(net) = \frac{2}{1 + e^{-\lambda net}} - 1 \tag{3.5}$$

3) Log-sigmoid

$$f(net) = \frac{1}{1 + e^{-\lambda net}} \tag{3.6}$$

4) Linear

$$f(net) = net \tag{3.7}$$

For simplicity of notation the following substitution may be used

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \\ -1 \end{bmatrix}, \quad W = \begin{bmatrix} w_1 \\ \vdots \\ w_n \\ w_{n+1} \end{bmatrix}. \tag{3.8}$$

So, the input-output relation of a neuron can be written as

$$y = f(net) = f(W'x) \tag{3.9}$$

where the prime (') represents the transpose of a matrix.

3.2.4 Learning Algorithms

A learning algorithm (or training algorithm) is an algorithm by which the neural network stores new information. It may be applied by the neural network itself (unsupervised learning) or by a supervisory system (supervised learning). The learning may take place by (i) developing new connections, (ii) losing some existing connections inside the network, and/or (iii) modifying the internal parameters, especially internal weights. The last modification (iii) has been paid the most attention.

Training a neural network in this study involves only the modification of the connection weights within the network. In other words the training algorithm performs a search through the space of neural network parameters (here only connections of neurons not the parameters of activation functions) in order to find a set of parameters with which the neural network satisfies the condition or performs the required function. The most frequently used algorithm is Rumelhart's error backpropagation [19].

3.2.5 Operation Schedule

The operation schedule is a list of times and the neurons which become active at those instants of time. A categorization of the operation schedule is top-down, bottom-up, and interactive [20]. For the first two categories, top-down and bottom-up, the neurons are grouped into layers, and higher layers (farther from the input) do not affect the lower layers (nearer to the neural network inputs). Neurons in every layer are numbered and they are activated sequentially. After all the neurons of a layer are activated the next layer of neurons is activated in the same manner. In a bottom-up schedule layers which are closer to the input are activated first.

In interactive models neurons of different layers (higher or lower) affect one another, so the information flows in both directions. A more restricted model of this type is when information flows only between adjacent levels. One of the schemes of activating neurons in such networks is activation in a random manner.

In a feedforward multilayer network the neurons are activated in a bottom-up schedule i.e. those which are closer to the input respond first. Their outputs feed into the neurons of the next layer, which then respond. In networks with feedback, the set of neurons responding at a given time may be chosen randomly, i.e. the interactive operation schedule can be used.

3.3 Training a Neuron

Training a neuron means adjusting w_i and $bias$ so that $f(net)$ is closer to some desired value than it was before training.

There are several learning rules which define how w_i should be changed to result in the desired change at the output of the network. Among them is delta learning rule which in error backpropagation training scheme for multi-layer neural networks. Since error backpropagation is used for the training of the neural networks in the neural network based decentralized controller, delta learning rule will be explained in the following.

Suppose that the desired output of the neuron is d and its actual output is y. The adjustment of w_i, $w_i \in W$, can be expressed as the following optimization problem.

$$\underset{W}{Min} \quad E = \frac{1}{2}(d-y)^2 \tag{3.10}$$

$$y = f(W'x) = f(net) \; . \tag{3.11}$$

Hence, the minimization problem can be rephrased as follows.

$$\underset{W}{Min} \quad E = \frac{1}{2}(d - f(W'x))^2 \tag{3.12}$$

The delta rule is actually the steepest descent solution of the optimization problem. In this method the scaled negative gradient of the function to be minimized, here E, is added to W. The gradient of E with respect to W is

$$\nabla E = -(d-y)f'(W'x)x \tag{3.13}$$

where f' is the derivative of f with respect to its argument and W^T is the transpose of W. So,

$$\triangle W = -\eta \ \nabla E \tag{3.14}$$

where η is called the learning rate. Hence, after training, the new weight is given by

$$W_{new} = W_{old} - \eta \nabla E \ . \tag{3.15}$$

The gradient of E requires the computation of $f'(W^T x)$ which is the slope of the activation function. The following presents $f'(W^T x)$ for those activation functions which will be used in our control problem.

1) linear activation function

$$f'(net) = 1 \tag{3.16}$$

2) tan-sigmoid activation function

$$\begin{aligned} f'(net) &= \frac{2\lambda e^{-\lambda net}}{1 - e^{-\lambda net}} \\ &= -\lambda f(net)[1 - f(net)] \end{aligned} \tag{3.17}$$

3.4 Error Backpropagation

In a multilayer neural network when an input is presented to the network the corresponding output will be generated by the network operation in a feedforward manner, using the bottom-up operation schedule. But, when in learning mode, the neural network weight adjustment propagates from the output layer to the input layer through hidden layers. This sequential backpropagating weight adjustment has inspired the name of this training algorithm. This method of training neural networks was initially introduced by Werbos [25] and later developed by Rumelhart and McClelland [20]. Backpropagation is the steepest decent algorithm of minimization and is a generalized form of the delta learning rule.

In order to study the backpropagation algorithm, let us consider the following two layer network:

$$\begin{array}{ccccccc} \bar{x} & \rightarrow & \Gamma_1[.] & \rightarrow & y & \rightarrow & \Gamma_2[.] \rightarrow z \\ & & W_1 & & & & W_2 \end{array} \tag{3.18}$$

where x is the input of the network and y and z represent the output of the first and the second layers of the neural network; $\Gamma_1[.]$ and $\Gamma_2[.]$ stand for the activation functions of the corresponding layer neurons; W_1 and W_2 are scaling matrices of appropriate dimensions.

The other notations are defined as follows.

$$\bar{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}_{m\times 1}, \quad \bar{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}_{n\times 1}, \quad \bar{z} = \begin{bmatrix} z_1 \\ \vdots \\ z_q \end{bmatrix}_{q\times 1} \tag{3.19}$$

and

$$\bar{y} = \Gamma_1[W_1\bar{x}], \quad \bar{z} = \Gamma_2[W_2 y]. \tag{3.20}$$

Let

$$W_2 = \begin{bmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & & \\ w_{q1} & \cdots & w_{qn} \end{bmatrix}_{q \times n} \tag{3.21}$$

and

$$\bar{net} = W_2 \bar{y} = \begin{bmatrix} net_1 \\ \vdots \\ net_q \end{bmatrix}_{q \times 1} . \tag{3.22}$$

Then

$$\Gamma_2[W'_2 \bar{y}] = \Gamma_2[\bar{net}] = \begin{bmatrix} f_2(net_1) \\ \vdots \\ f_2(net_q) \end{bmatrix}_{q \times 1} . \tag{3.23}$$

Updating an element of W'_2 based on the steepest descent algorithm is equivalent to computing

$$\triangle w_{kj} = -\eta \frac{\partial E}{\partial w_{kj}}. \tag{3.24}$$

Applying the derivative chain rule, we get

$$\triangle w_{kj} = -\eta . \frac{\partial (net_k)}{\partial w_{kj}} . \frac{\partial E}{\partial net_k} \tag{3.25}$$

where net_k is an element of $\bar{net}$. We may define the delta (δ) for the kth element of z as

$$\delta_z = -\frac{\partial E}{\partial net_k} = -\frac{\partial z_k}{\partial net_k} . \frac{\partial E}{\partial z_k}. \tag{3.26}$$

From the topology of the network we get

$$\frac{\partial (net_k)}{\partial w_{kj}} = y_j \quad . \tag{3.27}$$

Thus,

$$\Delta w_{kj} = \eta \delta_z y_j \quad . \tag{3.28}$$

The generalized form of the above equations which is valid for a layer, can be written completely as

$$\bar{\delta}_z = -\frac{\partial E}{\partial \bar{net}} = -\begin{bmatrix} \frac{\partial E}{\partial z_1} . f_2^*(net_1) \\ \vdots \\ \frac{\partial E}{\partial z_q} . f_2^*(net_q) \end{bmatrix}. \tag{3.29}$$

From

$$\bar{net} = W_2 \bar{y} \tag{3.30}$$

we can conclude that

$$\frac{\partial E}{\partial W_2} = \frac{E}{\partial net} . y'. \tag{3.31}$$

$$\frac{\partial E}{\partial \bar{y}} = \frac{\partial E}{\partial net} . W_2'. \tag{3.32}$$

Thus,

$$\Delta W_2 = \eta \delta_z \bar{y}' \tag{3.33}$$

$$\frac{\partial E}{\partial \bar{y}} = \bar{\delta}_z \ W_2'. \tag{3.34}$$

By repeating the above procedure $\frac{\partial E}{\partial W}$ can be found for the first layer of the neural network and the weight adjustment for the whole network can take place.

As mentioned earlier, the optimization technique which is used in the standard backpropagation technique is the steepest decent algorithm ([19], [26]). Some methods have been developed to improve the convergence property of this training scheme, ([27]-[31]). Also other optimization techniques such as quasi Newton ([32], [33]), and conjugate gradient ([34]) methods may also be used to provide better convergence properties ([35]).

3.5 Learning a Function using Neural Networks

Neural networks can learn a wide range of variable dependencies from deterministic relationships to stochastic relationships. What we are looking for in this study is a multivariable static mapping of control system variables. In order to train the neural network for a function of some variable(s), usually a set of desired output samples and the corresponding input variables are stored as training patterns. Then each input is presented to the neural network and its actual output is found. The difference between the actual output and the desired output, enables the training scheme to adjust the internal parameters of the neural network in such a way that the actual output after training is closer to the desired value than it was before training.

For example, suppose that a system is required to learn a nonlinear function, say

$$f(x) = sin(x). \tag{3.35}$$

By giving some samples of the independent variable x_i, i indicates the sample, and the corresponding function values $f(x_i)$, for example, a hundred equally spaced samples of a period and the corresponding function values, a neural network with symmetric sigmoid neurons, as shown below, can learn their relationship, within the period, [18].

Learning the relationship for Figure 3.5 means that some w_i, w_i', and $bias_i$ can be found such that the $out(x)$ is as close to $f(x)$ as is required. When the training is complete the output of the neural network can be as close to $f(x)$ as required at those values of x that $(x, f(x))$ were presented to the network [36], assuming that there is a sufficient number of neurons in the network.

As can be seen from the above example training a neural network for a continuous function has been performed as training the network for a set of discrete variables. This is valid only because a neural network interpolates the data that it

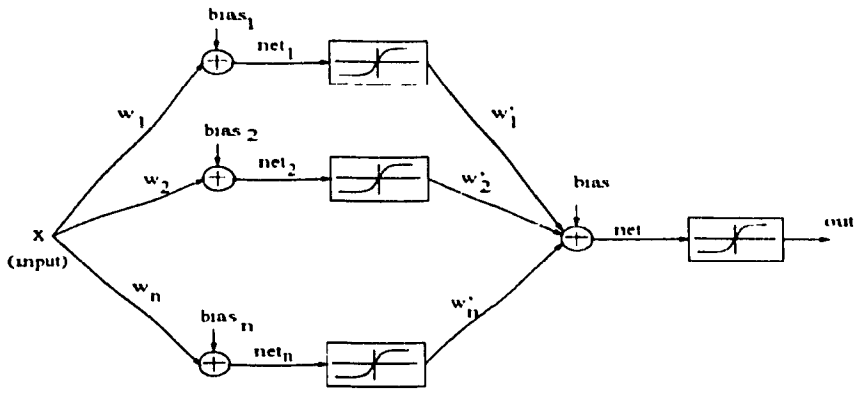

Figure 3.5: A multilayer neural network to learn a symmetric function. All neurons are of symmetric sigmoid type.

has learned. So, for those values of x in the example for which there has not been any training, the neural network gives an output which is a result of an interpolation of the known data; so we may expect that $out(x) \approx f(x)$.

Presenting samples of inputs and outputs is not the only way of making a neural network learn a function. For example when using the backpropagation technique, only the direction of change for the actual output towards the desired value is sufficient. So, in some applications of neural networks an estimate of this direction can be used for training purposes.

A method of learning a function which is usually used when dealing with dynamical systems involves a function of neural network independent inputs and their delays (MA model of dynamical systems), i.e.,

$$out(\bar{x}(t)) = f(\bar{x}(t), \bar{x}(t - \triangle t), \bar{x}(t - 2\triangle t), \cdots, x(t - n\triangle t)) \qquad (3.36)$$

We can also use a function of delayed inputs and outputs (ARMA model of dynamical systems) [37].

$$\begin{aligned} out(\bar{x}(t)) = {} & f(y(t - \triangle t), y(t - 2\triangle t), \cdots + y(t - m\triangle t) \qquad (3.37) \\ & + \bar{x}(t), \bar{x}(t - \triangle t), x(t - 2\triangle t), \cdots, x(t - n\triangle t)). \end{aligned}$$

Here f is a, nonlinear, function which is to be learned by the neural network.

The above models are standard ways of representing dynamical systems. The following figure depicts this form of modeling a system. In Figure 3.6, Δ is a

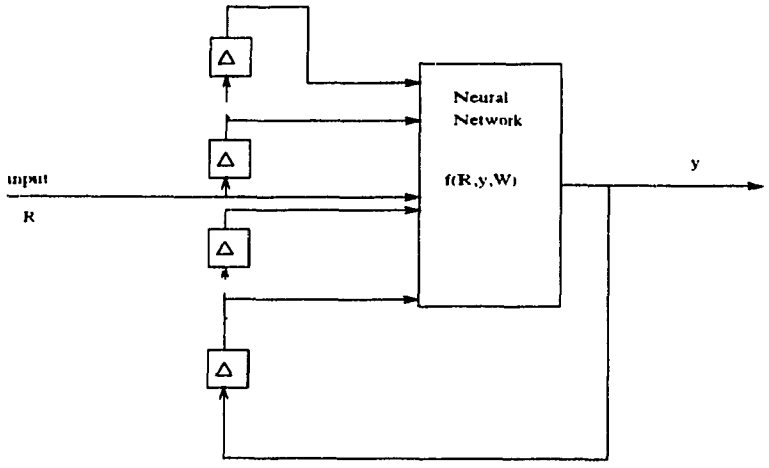

Figure 3.6: A neural network model of some nonlinear systems.

constant delay, for continuous systems, which can be replaced by z^{-1} for discrete systems.

3.6 Neural Networks and Optimization in Control Systems

In this section, we look at the problem of neural network training as an optimization problem. For this purpose it may be supposed that a set of independent variables x_i and the corresponding dependent variables $f(x_i)$ are given. A neural network N is supposed to learn f such that

$$N(x_i) \approx f(x_i). \tag{3.38}$$

The internal parameters of the neural network would be represented by W; so N is a function of x and W, i.e.,

$$N(x_i) = N(x_i, W). \tag{3.39}$$

The optimization problem is defined for minimizing some error or cost function. The most commonly used function is the quadratic error function

$$E = \sum_i (f(x_i) - N(x_i, W))^2 \tag{3.40}$$

So the problem of training the neural network is actually the following minimization problem

$$\underset{W}{Min}\ E = \sum_i (f(x_i) - N(x_i, W))^2. \tag{3.41}$$

In the following the backpropagation algorithm is considered as the training scheme.

We have seen that for finding $\frac{\partial E}{\partial w_i}$, $w_i \in W$, in the backpropagation technique we would need to have $\frac{\partial E}{\partial z_i}$, z_i an output of the neural network, and the status of the neural network. We may suppose that the supervisory system for this supervised learning problem takes in the derivative, or an estimate of the derivative, of E with respect to the output of the neural network, and it produces the derivative, or an

estimate of the derivative, of E with respect to the internal parameters of the neural network.

A generalization of this problem is that we consider the neural network in a system, say a control system, and try to minimize an energy or cost function, say J. If $\frac{\partial E}{\partial z_i}$, or an estimate of it, is given to the supervisory system it would produce the derivative, or an estimate of the derivative, of J with respect to the internal parameters of the network. Once the derivative of the energy, or cost, function with respect to some parameters (here network's internal parameters) is found, this optimization problem, which is now an optimal control problem, can be solved, e.g. see [38].

An important issue in incorporating neural networks in control systems is how to train them for the system. The minimization problem of equation 3.41 can be approached as follows. For every i, $f(x_i - N(x_i, W))$ is backpropagated through the neural network yielding $N(x_i, W')$ closer to $f(x_i)$. When all the samples have been used for training, one epoch is computed and the training can be started again from $i = 1$. This form of training, can be carried out until the value of E is less than some desired value.

In a control system, if the neural network is included in it, the required data for neural network training can be collected, the network trained and then incorporated in the system. This form of training a neural network is called off-line training. Not all control problems can be dealt with by using off-line training, especially when the network is supposed to be an active part of the control system. In this case, the neural network is incorporated in the system and it will be operational while being trained. This form of training is on-line training. A variety of different schemes are conceivable for the on-line training of a neural network. The following is one of them. Suppose that the neural network can be operated in discrete instants of time. Thus, there is an interval wherein the neural network can be trained, i.e., training and control can be applied one after the other. This scheme is particularly

suitable for slow systems like the MZSH system under consideration in this thesis. Epochwise training is also possible in on-line training. A set of pieces of data can be collected and assorted for training the network, assuming that the time interval for training allows the neural network training for more than one set of training data, and the set can be updated as the system evolves in time.

Chapter 4

The Neural Network Decentralized Controller

4.1 Introduction

In this chapter, neural network based decentralized control is introduced. This controller which is an output feedback decentralized controller has similar structure to that of controllers introduced in Chapter 2. While keeping the decentralized structure for the controller, we add neural networks to the control system. The neural network part adjusts the controller gains so that they are optimal for each setpoint and that the overall system has optimal gains over a wide range of operation, such as the typical daily range of operation of the heating system [1].

The purpose of this study is to investigate the possibility of using neural networks in an MZSH system, and the trainability of the networks in that system.

Hence, every aspect of the control system is intentionally kept as simple as possible. For instance, one neural network is assigned to each controller gain. The neural networks used are feedforward networks which perform a static mapping, and the optimization technique including neural network training is the steepest decent method. Although, this simplicity helps in understanding the problem, it gives rise to stability and global minimization issues.

The results presented at the end of this chapter show the capability of neural network based control for performing control action for multizone space heating systems.

This chapter is organized as follows: the time varying formulation of the control problem is presented in section 2. The neural network decentralized controller structure is presented in section 3. Section 4 formulates an approximation to the gradient of the cost function and shows how the steepest decent optimization technique is employed. Section 5 presents the simulation results which show the trainability of the neural networks, improvement of performance with respect to an optimal robust controller, performance of the controller in applications requiring a wide range of operation, disturbance rejection, and the use of different types of neural networks for the control action.

4.2 Time Varying Controller

As explained in Chapter 2 a decentralized output feedback controller can control the MZHS system almost as well as a centralized controller [9]. The design was based on a linear model of the plant, i.e.,

$$\Delta\dot{x}(t) = A\Delta x(t) + \sum_{i=1}^{3} B_i \Delta u_i(t) + E\Delta d(t) \tag{4.1}$$

$$\Delta y_i(t) = C_i \Delta x(t), \qquad 1 \le i \le 3 \tag{4.2}$$

where Δ stands for the change of variable from its setpoint value. A constant gain controller was used, i.e.,

$$u(t) = \tilde{K}^{(1)} e(t) + \tilde{K}^{(2)} \xi(t) \tag{4.3}$$

where,

$$\tilde{K}^{(1)} = \begin{bmatrix} \tilde{K}_5 & 0 & 0 \\ 0 & \tilde{K}_1 & 0 \\ 0 & \tilde{K}_7 & 0 \\ 0 & 0 & \tilde{K}_3 \\ 0 & 0 & \tilde{K}_9 \end{bmatrix}, \tilde{K}^{(2)} = \begin{bmatrix} \tilde{K}_6 & 0 & 0 \\ 0 & \tilde{K}_2 & 0 \\ 0 & \tilde{K}_8 & 0 \\ 0 & 0 & \tilde{K}_4 \\ 0 & 0 & \tilde{K}_{10} \end{bmatrix}. \tag{4.4}$$

This constant gain controller is optimal at one constant setpoint. By rearranging the model of the HVAC system in Chapter 2 the following model is obtained.

$$\dot{x}(t) = \tilde{A}x(t) + \sum_{i=1}^{3} \tilde{B}_i(t) u_i(t) + \tilde{E}d(t) \tag{4.5}$$

$$y_i(t) = \tilde{C}_i x(t), \qquad 1 \le i \le 3. \tag{4.6}$$

where $\tilde{B}_i(t)$ is actually $\tilde{B}_i(x(t))$. An optimal controller which can best minimize the cost function

$$J(K^{(1)}, K^{(2)}) = \int_0^t (e'(t)Qe(t) + u'(t)Ru(t))dt \tag{4.7}$$

when e and u are related to a time-varying system should be time-varying (e.g. see Chapter 2). The time-varying controller which is proposed in this study consists of gains with two terms; a constant gain and a time-varying gain. The constant term

provides a stable controller if the time-varying term is set equal to zero. The time varying term will be found such that it improves the performance of the control system. Consider the following controller

$$K^{(1)}(t) = N^{(1)}(t) + \tilde{K}^{(1)} \quad , \quad K^{(2)}(t) = N^{(2)}(t) + \tilde{K}^{(2)} \tag{4.8}$$

Where $\tilde{K}^{(1)}$ and $\tilde{K}^{(2)}$ are the same as those defined in equations 4.3 and 4.4. In order to keep the decentralized structure of the controller, the zero elements of $\tilde{K}^{(1)}$ and $\tilde{K}^{(2)}$ have zero valued counterparts in $N^{(1)}(t)$ and $N^{(2)}(t)$, and the nonzero elements of $\tilde{K}^{(1)}$ and $\tilde{K}^{(2)}$ have counterparts in $N^{(1)}(t)$ and $N^{(2)}(t)$ which are the outputs of the neural networks. Therefore, ten neural networks are incorporated in the control system, i.e., $K^{(1)}$ and $K^{(2)}$: are defined as follows.

$$K^{(1)} = \begin{bmatrix} N_5(t) + \tilde{K}_5 & 0 & 0 \\ 0 & N_1(t) + \tilde{K}_1 & 0 \\ 0 & N_7(t) + \tilde{K}_7 & 0 \\ 0 & 0 & N_3(t) + \tilde{K}_3 \\ 0 & 0 & N_9(t) + \tilde{K}_9 \end{bmatrix}, \tag{4.9}$$

and

$$K^{(2)} = \begin{bmatrix} N_6(t) + \tilde{K}_6 & 0 & 0 \\ 0 & N_2(t) + \tilde{K}_2 & 0 \\ 0 & N_8(t) + \tilde{K}_8 & 0 \\ 0 & 0 & N_4(t) + \tilde{K}_4 \\ 0 & 0 & N_{10}(t) + \tilde{K}_{10} \end{bmatrix}. \tag{4.10}$$

Each neural network is a static mapping of the signals available in the control system. Thus, the time dependency of $N_i(t)$ is initiated by the time variation of its inputs. The sufficient neural network input variables were found to be the output ($\bar{y}(t)$), the output error ($\bar{e}(t)$), and the integral error ($\dot{\xi}(t)$). Therefore, the neural network functions were defined as

$$N_i(t) = N_i(\bar{y}(t), \bar{e}(t), \dot{\xi}(t)). \tag{4.11}$$

The decentralized structure of the controller restricts the above inputs to the signals available at the station corresponding to the network. The following equations present the complete definition of the controller gain matrices:

$$K^{(1)} = \begin{bmatrix} N_5(e_1(t),\xi_1(t),y_1(t),\nu_3(t)) + \tilde{K}_5 & 0 & 0 \\ 0 & N_1(e_2(t),\xi_2(t),y_2(t),\nu_1(t),\nu_4(t)) + \tilde{K}_1 & 0 \\ 0 & N_7(e_2(t),\xi_2(t),y_2(t),\nu_1(t),\nu_4(t)) + \tilde{K}_7 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & N_3(e_3(t),\xi_3(t),y_3(t),\nu_2(t),\nu_5(t)) + \tilde{K}_3 \\ 0 & 0 & N_9(e_3(t),\xi_3(t),y_3(t),\nu_2(t),\nu_5(t)) + \tilde{K}_9 \end{bmatrix}, \quad (4.12)$$

and

$$K^{(2)} = \begin{bmatrix} N_6(e_1(t), \xi_1(t), y_1(t), \nu_3(t)) + \tilde{K}_6 & 0 \\ 0 & N_2(e_2(t), \xi_2(t), y_2(t), \nu_1(t), \nu_4(t)) + \tilde{K}_2 \\ 0 & N_8(e_2(t), \xi_2(t), y_2(t), \nu_1(t), \nu_4(t)) + \tilde{K}_8 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & N_4(e_3(t), \xi_3(t), y_3(t), \nu_2(t), \nu_5(t)) + \tilde{K}_4 \\ 0 & N_{10}(e_3(t), \xi_3(t), y_3(t), \nu_2(t), \nu_5(t)) + \tilde{K}_{10} \end{bmatrix}. \quad (4.13)$$

The neural networks are set such that their outputs are zero at the beginning of training, i.e., $N_i(0)$ is equal to zero for $1 \leq i \leq 10$. $\tilde{K}_{(1)}$ and $\tilde{K}_{(2)}$ are assumed to be able to provide stability for the control system, so that the control system is stable when the training starts. The controller starts from $K^{(1)}(t) = \tilde{K}^{(1)}$ and $K^{(2)}(t) = \tilde{K}^{(2)}$, and over time the neural network training provides $N^{(1)}(t)$ and $N^{(2)}(t)$ such that the control system's operation with $K^{(1)}(t) = N^{(1)}(t) + \tilde{K}^{(1)}$ and $K^{(2)}(t) = N^{(2)}(t) + \tilde{K}^{(2)}$ generates a lower value of the cost (J) than its operation with $K^{(1)}(t) = \tilde{K}^{(1)}$ and $K^{(2)}(t) = \tilde{K}^{(2)}$.

4.3 Neural Network Controller

Neural networks as the time-varying terms of controller gains start their operation with the output values equal to zero. The training of the neural networks uses the steepest descent approach. The gradient minimization is as follows.

Suppose f is a function of some variables, say $\theta_1, \cdots, \theta_n$. We define the (Jacobian) gradient, $\frac{\partial f}{\partial \bar{\theta}}$, of f with respect to the vector $\bar{\theta} \in R^n$ as follows [39]

$$\frac{\partial f}{\partial \bar{\theta}} = [\frac{\partial f}{\partial \theta_1} \frac{\partial f}{\partial \theta_2} \cdots \frac{\partial f}{\partial \theta_n}]' \tag{4.14}$$

where

$$\bar{\theta} = [\theta_1, \theta_2, \cdots, \theta_n]' \tag{4.15}$$

and the prime ($'$) represents the transpose of a vector. If the operating point, in the parameter space, is changed from $\bar{\theta}_{nom}$ to $\bar{\theta}_{nom} + \Delta\bar{\theta}$, where

$$\Delta\bar{\theta} = -\eta \frac{\partial f}{\partial \bar{\theta}} \tag{4.16}$$

$$0 < \eta \ll 1 \tag{4.17}$$

if η is sufficiently small then it follows that

$$f(\bar{\theta}_{nom} + \Delta\bar{\theta}) \leq f(\bar{\theta}_{nom}). \tag{4.18}$$

The value of η may also be found by a one-dimensional search. Normally in the backpropagation technique, the function to be minimized is

$$E = \frac{1}{2}\sum_i (d_i - out_i)^2 \tag{4.19}$$

where out_i is the actual output of the network for an input and d_i is the corresponding desired output. In order to minimize E, each term of it i.e. $(d_i - out_i)$ will be minimized sequentially. In order to minimize one term of E, say the jth element

$$-\frac{\partial E}{\partial out_j} = (d_j - out_j) \tag{4.20}$$

,j is a specific value of i, is backpropagated through the network. Thus, the output of the neural network (out) to that input, after training, is closer to the desired value d_i than the output before training. In other words, backpropagating $-\frac{\partial E}{\partial out}$ adjusts the internal parameters of the neural network such that

$$E_{after\ training} \leq E_{before\ training} \tag{4.21}$$

The same concept is used in minimization of an arbitrary cost function, say ,J in the case of a dynamical system. The value of $-\frac{\partial J}{\partial out}$ is found and is backpropagated through the neural network. Finding the gradient of the cost function is discussed in the next section.

Suppose that the controller gain at an instant can be represented by a function of some of the signals in the control system. Applying the gradient method, which is introduced above, makes the controller gain closer to its optimal value anytime the optimization is performed. Thus, if the control system reaches the same condition next time its performance is improved.

For the control system in this study, it is assumed that the three signals y, ϵ, and $\bar{\xi}$ are sufficient inputs for the controller. The mapping of these signals to the gain is taken care of by using a three-layer neural network. Figure 4.1 depicts the network.

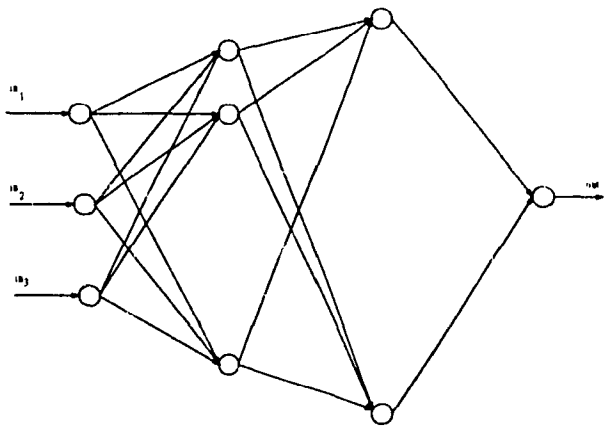

Figure 4.1: The neural network used in the control system.

The neural network has twenty neurons in the first layer, fifty neurons in the second layer and one neuron in the third layer. Since $N_t(t)$ may take positive or negative values, all the neurons are chosen to have symmetric activation functions. The output neuron of each network cannot be of saturation type because the proper gain range is not known *a priori*. The neurons are chosen to be tan-sigmoid for the first and the second layers and linear for the output layer. Weights are initialized to small numbers using MATLAB's random number generator with a scale of 0.1. The bias of the output neuron is calculated such that the network's output is zero for the initial condition of the control system. The scaled negative gradient of the cost function is backpropagated through the neural network. A scale of about 0.01 on the gradient and a learning rate of 0.01 were found to provide stability and training.

In order to prevent possible violation of the bounds af heating system control inputs, the controller output is limited by a hard limiter whose threshold levels are set at about the heating system input limits. Anytime each limiter is active it switches on a condition in the appropriate neural network in order to train the network to prevent saturation.

Figure 4.2 illustrates the control system structure.

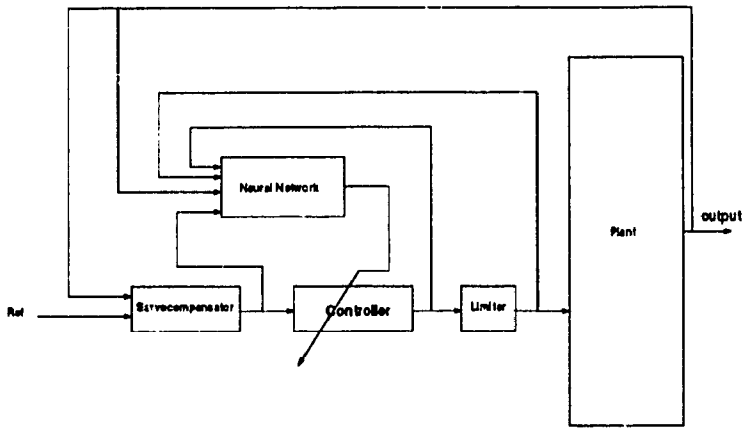

Figure 4.2: Neural network control system.

4.3.1 The Derivative of Vectors

In order for finding $\frac{\partial J}{\partial \bar{\omega}}$, the derivative of a scalar and a vector with respect to a vector and the chain rule for differentiation are needed. The notation that is used here is compatible with [39].

If all vectors are column vectors,

1) For $J = R$ and $\bar{\omega} = R^{m\times 1}$ the derivative of J with respect to ω is

$$\frac{\partial J}{\partial \bar{\omega}} = \begin{bmatrix} \frac{\partial J}{\partial \omega_1} \\ \frac{\partial J}{\partial \omega_1} \\ \vdots \\ \frac{\partial J}{\partial \omega_m} \end{bmatrix}_{m\times 1} . \tag{4.22}$$

$\frac{\partial J}{\partial \bar{\omega}}$ is the Jacobian gradient of J with respect to the m parameters of ω. The Jacobian gradient is the transpose of ∇J, which is normally used in optimization [40].

2) If $x \in R^{n\times 1}$ and $y \in R^{m\times 1}$ are two vectors, then the derivative of x with respect to y is given by

$$\frac{\partial \bar{y}}{\partial \bar{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_2}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_1} \\ \frac{\partial y_1}{\partial x_2} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_2} \\ & \vdots & & \\ \frac{\partial y_1}{\partial x_n} & \frac{\partial y_2}{\partial x_n} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}_{n\times m} \tag{4.23}$$

where $y_1, y_2, \cdots, y_m$ and $x_1, x_2, \cdots, x_n$ are the elements of y and x respectively.

3) The chain rule for the differentiation of vectors is

$$\frac{\partial \bar{z}}{\partial \bar{x}} = \frac{\partial \bar{y}}{\partial \bar{x}} \frac{\partial z}{\partial \bar{y}} \tag{4.24}$$

which for $\bar{z} \in R^{r\times 1}$ gives

$$\left[\frac{\partial \bar{z}}{\partial \bar{x}}\right]_{n\times r} = \left[\frac{\partial \bar{y}}{\partial \bar{x}}\right]_{n\times m} \left[\frac{\partial \bar{z}}{\partial y}\right]_{m\times r} \tag{4.25}$$

4.3.2 Gradient of the Cost Function

The general form of the cost function to be minimized is

$$J = \int_0^t [\bar{e}'(t)\, Q\, \bar{e}(t) + \bar{u}'(t)\, R\, \bar{u}(t)]dt \tag{4.26}$$

where

$$e_i(t) = y_{r_i}(t) - y_i(t) \tag{4.27}$$

with the feedback relation

$$\bar{u}(t) = K \begin{bmatrix} \bar{e}(t) \\ \bar{\xi}(t) \end{bmatrix} \tag{4.28}$$

where

$$\bar{u}(t) \equiv \begin{bmatrix} u_1(t) \\ u_2(t) \\ u_3(t) \end{bmatrix} \tag{4.29}$$

and

$$u_1(t) = \nu_3(t), \quad u_2(t) = \begin{bmatrix} \nu_1(t) \\ \nu_4(t) \end{bmatrix}, \quad u_3(t) = \begin{bmatrix} \nu_2(t) \\ \nu_5(t) \end{bmatrix}; \tag{4.30}$$

$\nu_i(t)$ are the normalized inputs to the plant whose maximum values are given in Table 2.1. We may also write

$$\bar{u}(t) = K^{(1)}\bar{e}(t) + K^{(2)}\bar{\xi}(t) \tag{4.31}$$

or

$$\begin{bmatrix} \begin{bmatrix} \nu_1(t) \\ \nu_4(t) \end{bmatrix} \\ \begin{bmatrix} \nu_2(t) \\ \nu_5(t) \end{bmatrix} \\ \nu_3(t) \end{bmatrix} = \begin{bmatrix} k_5 & 0 & 0 \\ 0 & k_1 & 0 \\ 0 & k_7 & 0 \\ 0 & 0 & k_3 \\ 0 & 0 & k_9 \end{bmatrix} \begin{bmatrix} e_1(t) \\ e_2(t) \\ e_3(t) \end{bmatrix} + \begin{bmatrix} k_6 & 0 & 0 \\ 0 & k_2 & 0 \\ 0 & k_8 & 0 \\ 0 & 0 & k_4 \\ 0 & 0 & k_{10} \end{bmatrix} \begin{bmatrix} \xi_1(t) \\ \xi_2(t) \\ \xi_3(t) \end{bmatrix} \tag{4.32}$$

$K^{(1)}$ presents the proportional gains, $K^{(2)}$ presents the integral gains, and $K = [K^{(1)}|K^{(2)}]$; Q and R are defined as follows

$$Q = q\ I_{3\times 3}, \quad R = r\ I_{5\times 5}. \tag{4.33}$$

In order to find the Jacobian gradient of J in the parameter space of K, we put all nonzero elements of K in a vector, say $\bar{\omega}$. Thus,

$$\bar{\omega} = [k_1\ k_2\ \cdots\ k_{10}]' \tag{4.34}$$

Then, the Jacobian gradient of J in feedback gain parameter space is shown by

$$\begin{aligned}\frac{\partial J}{\partial \bar{\omega}} &= \int_0^t \{\left[\frac{\partial \bar{e}(t)}{\partial \bar{\omega}}\right][Q\ \bar{e}(t) + Q'\ \bar{e}(t)] + \left[\frac{\partial \bar{u}(t)}{\partial \bar{\omega}}\right][R\ u(t) + R'\ u(t)]\}dt \\ &= 2\int_0^t \{\left[\frac{\partial \bar{e}(t)}{\partial \bar{\omega}}\right]\ Q\ \bar{e}(t) + \left[\frac{\partial \bar{u}(t)}{\partial \bar{\omega}}\right]\ R\ \bar{u}(t)\}dt \end{aligned} \tag{4.35}$$

and

$$\frac{\partial \bar{e}(t)}{\partial \bar{\omega}} = \frac{\partial \bar{u}(t)}{\partial \bar{\omega}}\ \frac{\partial \bar{e}(t)}{\partial \bar{u}(t)}\ . \tag{4.36}$$

In the above equations we have

$$\bar{\omega} = [k_i]_{10\times 1}, \quad \frac{\partial J}{\partial \bar{\omega}} = [\frac{\partial J}{\partial k_i}]_{10\times 1} \tag{4.37}$$

$$\frac{\partial \bar{e}(t)}{\partial \bar{u}(t)} = [\frac{\partial e_i(t)}{\partial \nu_j(t)}]_{5\times 3} \tag{4.38}$$

$$\frac{\partial \bar{u}(t)}{\partial \omega} = \left[\frac{\partial \nu_i(t)}{\partial k_j}\right]_{10\times 5} \tag{4.39}$$

The complete matrices are as follow.

$$\frac{\partial \bar{e}(t)}{\partial \bar{u}(t)} = \begin{bmatrix} \frac{\partial e_1(t)}{\partial \nu_1(t)} & \frac{\partial e_2(t)}{\partial \nu_1(t)} & \frac{\partial e_3(t)}{\partial \nu_1(t)} \\ \frac{\partial e_1(t)}{\partial \nu_4(t)} & \frac{\partial e_2(t)}{\partial \nu_4(t)} & \frac{\partial e_3(t)}{\partial \nu_4(t)} \\ \frac{\partial e_1(t)}{\partial \nu_2(t)} & \frac{\partial e_2(t)}{\partial \nu_2(t)} & \frac{\partial e_3(t)}{\partial \nu_2(t)} \\ \frac{\partial e_1(t)}{\partial \nu_5(t)} & \frac{\partial e_2(t)}{\partial \nu_5(t)} & \frac{\partial e_3(t)}{\partial \nu_5(t)} \\ \frac{\partial e_1(t)}{\partial \nu_3(t)} & \frac{\partial e_2(t)}{\partial \nu_3(t)} & \frac{\partial e_3(t)}{\partial \nu_3(t)} \end{bmatrix} \tag{4.40}$$

$$\frac{\partial \bar{u}(t)}{\partial \bar{\omega}} = \begin{bmatrix} \frac{\partial \nu_1(t)}{\partial k_1(t)} & \frac{\partial \nu_4(t)}{\partial k_1(t)} & \frac{\partial \nu_2(t)}{\partial k_1(t)} & \frac{\partial \nu_5(t)}{\partial k_1(t)} & \frac{\partial \nu_3(t)}{\partial k_1(t)} \\ \frac{\partial \nu_1(t)}{\partial k_2(t)} & \frac{\partial \nu_4(t)}{\partial k_2(t)} & \cdots & & \frac{\partial \nu_3(t)}{\partial k_2(t)} \\ & & \vdots & & \\ \frac{\partial \nu_1(t)}{\partial k_{10}(t)} & \frac{\partial \nu_4(t)}{\partial k_{10}(t)} & \cdots & & \frac{\partial \nu_3(t)}{\partial k_{10}(t)} \end{bmatrix} \tag{4.41}$$

To find the effect of any variation in an element of $K^{(1)}$ or $K^{(2)}$, we will consider the equation of an input to the plant.

$$\bar{u}(t) = K^{(1)}\bar{e}(t) + K^{(2)} \int \bar{e}(t)\, dt \tag{4.42}$$

$$\nu_i(t) = k_m e_j(t) + k_n \xi_j(t) \tag{4.43}$$

where k_m and k_n are the elements of $K^{(1)}$ and $K^{(2)}$ respectively, and

$$\bar{\xi}(t) = \int \bar{e}(t)\, dt \tag{4.44}$$

and

$$1 \leq i \leq 5, \quad 1 \leq j \leq 3. \tag{4.45}$$

Thus,

$$\frac{\partial \nu_i(t)}{\partial k_m} = e_j(t) \quad and \quad \frac{\partial \nu_i(t)}{\partial k_n} = \xi_j(t) \tag{4.46}$$

The complete equations are given in Appendix A.

For finding the elements of $\frac{\partial \bar{e}(t)}{\partial \bar{u}(t)}$ it can be considered that

$$\frac{\partial c_i(t)}{\partial \nu_j(t)} = -\frac{\Delta y_i(t)}{\Delta \nu_j(t)} \qquad 1 \leq i \leq 3, \;\; 1 \leq j \leq 5 \tag{4.47}$$

i.e., instead of finding the derivative of an output with respect to a plant input, the steady state response changes with respect to the plant's input changes can be employed. This is a multivariable form of what is suggested in [41]. Since heating systems usually work close to their steady state conditions such an approximation is

realistic. As it is shown in the Figures 4.3-4.5 it can be assumed that the derivatives are constant about the setpoint.

This substitution is particularly suitable for small changes at reference inputs (temperature adjustments by occupants). For large reference input changes (occupied/unoccupied cycle switch), which forces the plant's inputs to change a great deal - usually their entire range - the input range was divided into five parts, and for each part, the corresponding steady-state response changes with respect to input changes were measured. Figures 4.3-4.5 depict the change of output steady-state values with respect to input changes for the setpoint of

$$\nu_1 = 0.4, \quad \nu_2 = 0.45, \quad \nu_3 = 0.5, \quad \nu_4 = 0.5, \quad \nu_5 = 0.5 \tag{4.48}$$

i.e. the inputs are kept constant but the one which is shown in the fingures. Therefore, the derivative of the outputs of the plant with respect to its inputs would be replaced by the slope of the graphs shown in Figures 4.3-4.5.

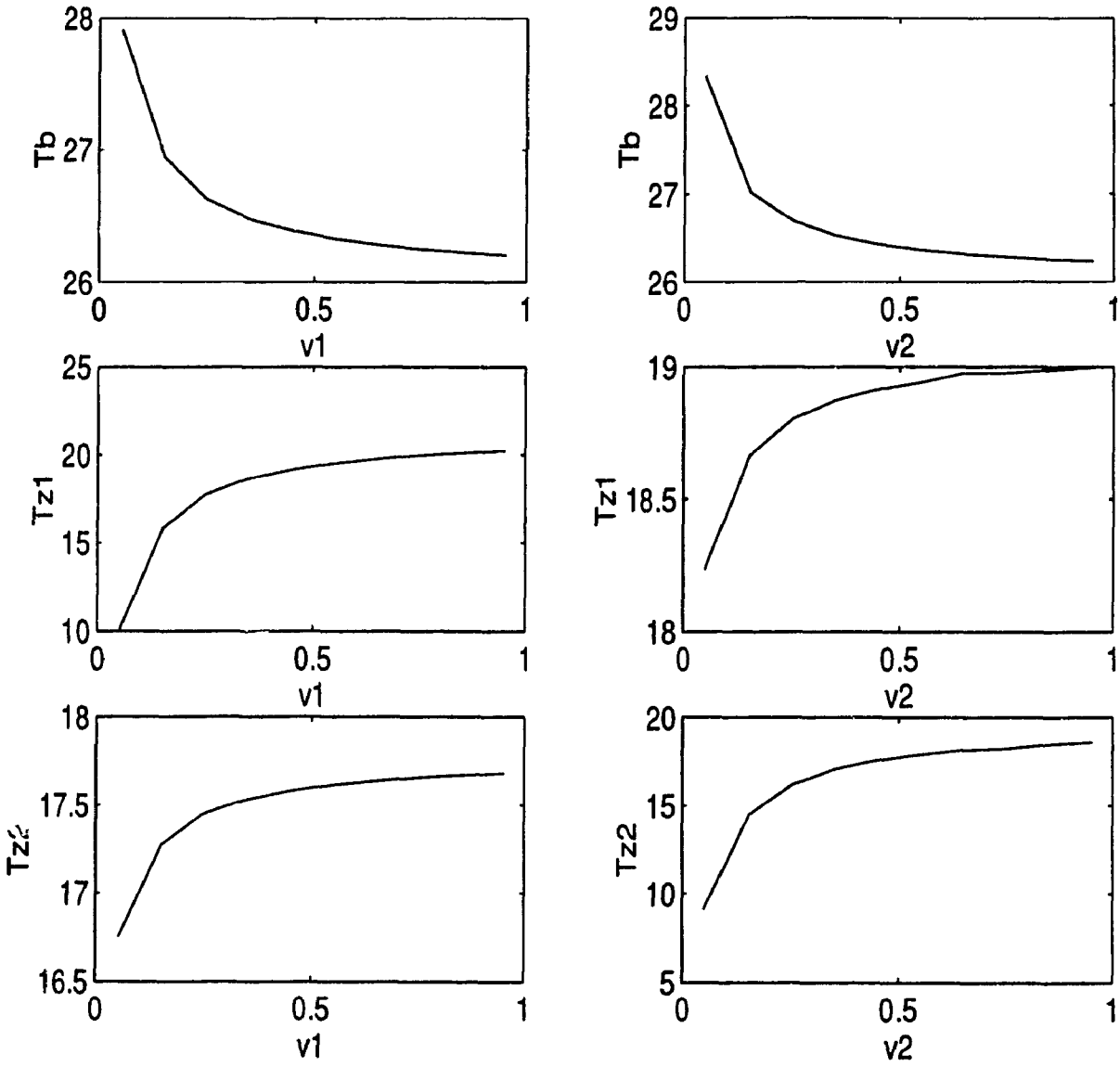

Figure 4.3: Steady state response of the plant when ν_1 and ν_2 change.

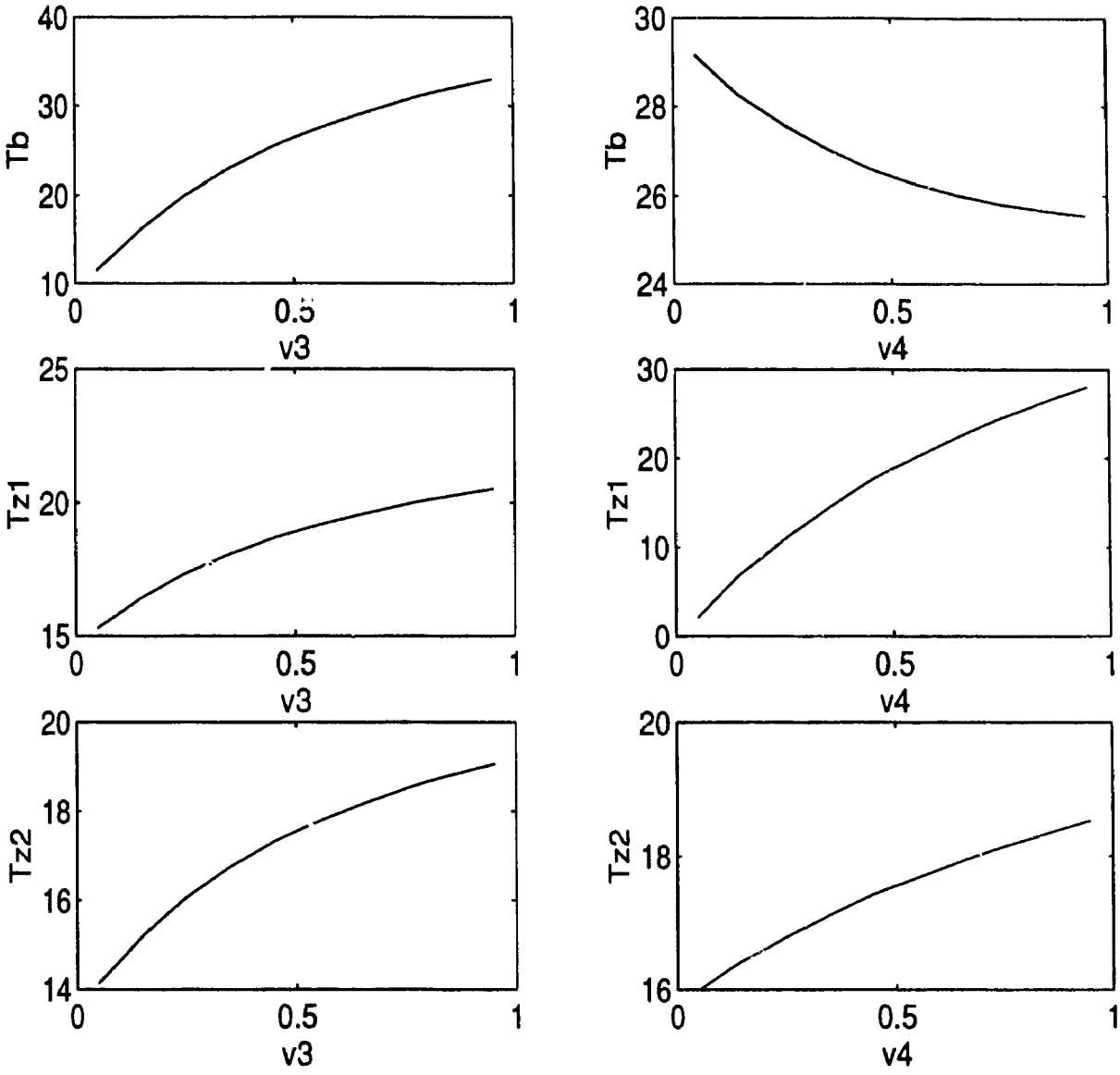

Figure 4.4: Steady state response of the plant when ν_3 and ν_4 change.

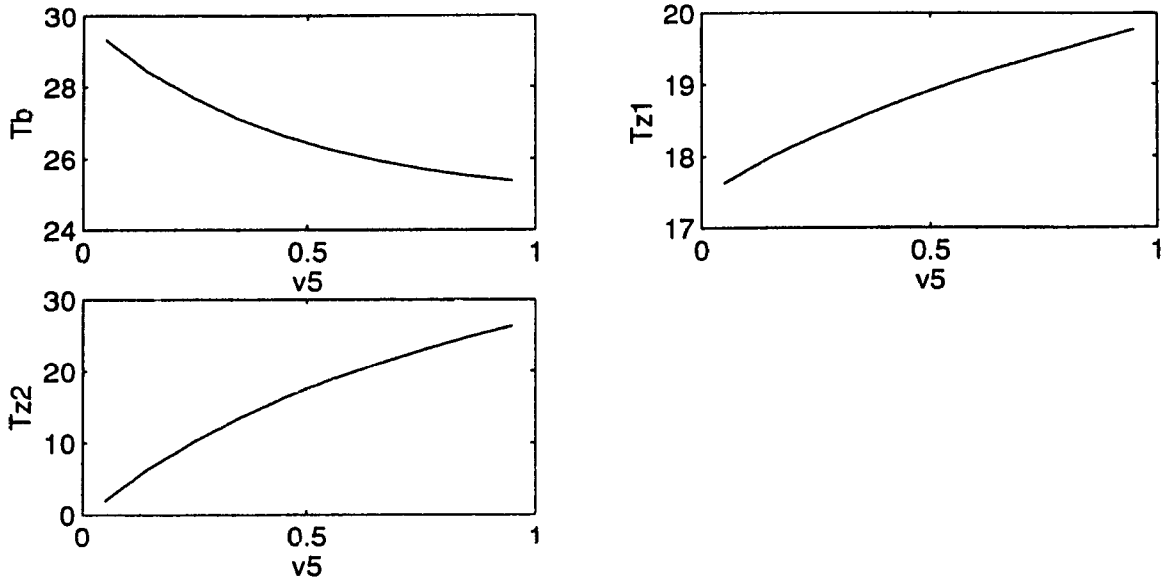

Figure 4.5: steady state response of the plant when ν_5 changes.

4.3.3 Decentralized Cost Minimization

The decentralized structure of the controller also raises the motivation of dividing the cost function into station costs. In our case, this is

$$J = J_1 + J_2 + J_3, \tag{4.49}$$

where

$$J_1 = \int \left[q\ \epsilon_1^2(t) + r\ u_1'(t).u_1(t)\right] dt, \tag{4.50}$$

$$J_2 = \int \left[q\ \epsilon_2^2(t) + r\ u_2'(t).u_2(t)\right] dt, \tag{4.51}$$

$$J_3 = \int \left[q\ \epsilon_1^2(t) + r\ u_3^2(t)\right] dt, \tag{4.52}$$

with $q = 1$ and $r = 10^{-5}$. In each station the gradients of the appropriate cost function with respect to the station's controller gains are found, and the corresponding neural networks are trained. This simplification, significantly reduces the computation effort.

4.4 Simulation Results

This section presents the result of the simulations carried out to investigate the performance of the neural network based controller under different operating conditions, and to compare its performance with that of the decentralized robust controller resulting from the linear design presented earlier. The linear decentralized robust controller gives performance comparable to that of the centralized robust controller also resulting from a linear design [9]. So, the decentralized controller is used as a measure of performance for the neural network based decentralized controller.

The simulations are categorized into variations about a constant setpoint, occupied/unoccupied operation, disturbance rejection, and different ways of recall.

1) Variation about a constant setpoint.

This simulation tests the performance of the neural network based controller when all its setpoints (or reference inputs) change in the form of pulse trains about some fixed main setpoint. From a practical perspective this status of operation simulates setpoint changes which could potentially be created by occupants when a building is occupied. So, better performance of one controller over another shows up as faster response to the imposed setpoint changes, and generally implies that the controller provides greater comfort for the occupants. This form of operation has other advantages which will be explained in detail later.

2) Occupied/unoccupied operation.

Saving energy in buildings is an improtant consideration because the inside temperature of a building does not need to be as high at night as it has to be during the day. The temperature can be switched from day-time value (occupied building temperature) to its night-time value (unoccupied building temperature) and vice versa. This type of saving raises three issues from the control point of view.

The first point is that for the designs which are based on linearized model, the model used would most likely not be valid over the whole range of operation. Therefore, a supervisory controller is needed in order to adjust the controller gains.

The second point is that this type of temperature setback requires large changes in setpoint values, or large changes of reference inputs, which may cause saturation at the inputs to the plant, and which show up the nonlinearities in the plant. Avoiding saturation in linear model based controllers potentially takes away some of performance ability of the heating system. The third point is that if the control system can cover the whole temperature range rapidly enough, the temperature of the building can be kept at unoccupied temperature for larger periods, therefore saving more energy.

3) Disturbance rejection.

Disturbance rejection in this heating system is the response of the control system to outdoor temperature changes which can affect the zone temperatures the most. For this test a profile of a normal winter's day in Montreal is applied to the control system while the setpoints are kept constant. In [9] it was shown that even a constant feedback gain controller provides good disturbance rejection. Here the response of the neural network controller is compared with the response of the robust controller to the same out-door temperature changes.

4) Different ways of recall.

The time variation of the controller parameters are tested in this part. Since on-line control requires a neural network's internal parameters to be continuously changing, the effect of slowing down the changes on the response of the control system is tested.

In another test the effect of slowing down the rate of change of the controller gains is studied. Both show a degradation of response, but the responses are still of acceptable quality.

4.4.1 Computer Facility Used in This Study

The environment of the required simulations in this study was accomplished using SIMULINK pakage of MATLAB. The Computation was carried out by SUN SPARC

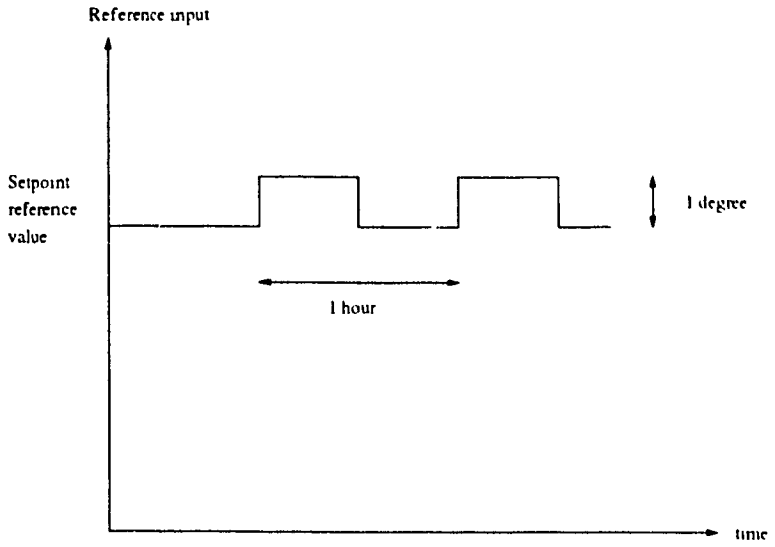

Figure 4.6: Reference input for neural network training

work stations (usually SPARK 10). The error tolerance of computations were between 10^{-5} and 10^{-11} depending on the simulations and time step for simulating the continuous systems. The time steps were mostly chosen to be between 0.001 hour and 0.0001 hour. The numerical computation method was Gear method. The time base of the discrete systems were chosen to be 0.01 hour which is in a matter of 100 times better than the time constant of the heating system.

4.4.2 Variation About a Constant Setpoint

The training was done for periodic reference inputs which were in the form of pulse variations about a set point. This reference variation has the following advantages:

1. It is similar to the change of setpoint by occupants

2. It gives the chance to train the neural networks about the setpoint and compare the response with that of the robust controller. Small variation of reference inputs about some nominal values keeps the states of the control system,

including the plant itself, in the neighborhood of their nominal values. Since the robust decentralized controller design, because of using the linearized model of the plant, is valid only if the states of the control system are in the neighborhood of their nominal values (see Chapter 2), this simulation provides a valid condition for a robust decentralized controller to operate thus to be compared with the neural network controller.

3. Since the reference inputs are periodic (with a period of 1 hour), after t=1 hr., a plot of the following cost function

$$J = \int_{(t-1)}^{t} [\bar{e}'(t)\, Q\, \bar{e}(t) + \bar{u}'(t)\, R\, \bar{u}(t)]\, dt \tag{4.53}$$

reveals whether or not the optimization is taking place. This part of the actual cost function shows the cost of one operation hour. So, at the end of each cycle this function shows the cost corresponding to that cycle. The comparison starts when the transient response is over, i.e. after t=2 hours.

Comparison of Costs

This simulation is done to see whether the minimization of cost by using the proposed controller is taking place. The result depicted in Figure 4.7 shows that as the neural network training continues the cost is reduced. Having a lower cost means better reference input tracking, in general.

The robust controller which is employed in this simulation is designed for this particular setpoint and is valid for the range of operation [9]. Since the design of this controller was based on the minimization of the same cost function as that of the neural network controller, the costs of operation for both neural network based and robust controller can be presented in the same figure.

The neural network based controller starts with some random weight matrices. Its output is used to adjust the gains of an already stable, but not optimal, controller.

Consecutive simulations, each presenting a lower cost uses the final neural network weight matrices of the preceding simulation. The learning rate of the neural network is kept constant in all simulations, which is of course, not the best way of doing the optimization.

The operation cost of the consecutive neural network based controller shows continuation of the cost minimization in general. Therefore, it can be concluded that the proposed controller is satisfying its goal.

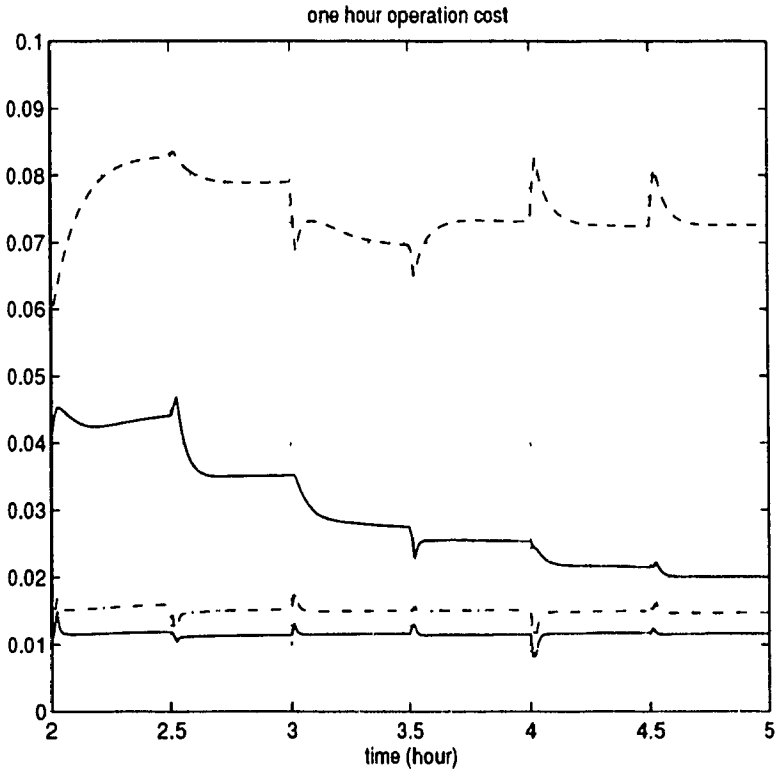

Figure 4.7: Comparison of costs for the robust controller and the neural network based controller. From top to bottom: robust controller, NN controller with 2 hours, 7 hours, and 15 hours of on-line training.

Responses to the Reference Inputs (with Variation about a Setpoint)

The response of the neural network control system to setpoint changes improves as training progresses. This improvement in the overall responses of the three stations to command signals is gradual. The comparison of the cost for the third simulation of the neural network based controller with the cost for the robust controller presents the outcome of the entire training. Although, the response of the neural network based controller (Figure 4.8-4.10) seems slow when the output is close to the setpoint value, it improves with further training.

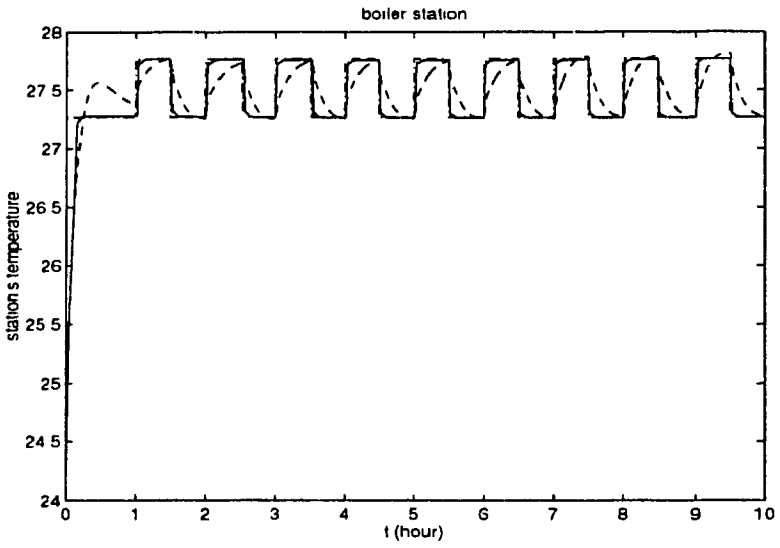

Figure 4.8: Station 1, comparison of the time-domain response of the NN based controller (solid line) and of the robust controller (dashed line). The dashdot line is the setpoint (almost matching the solid line).

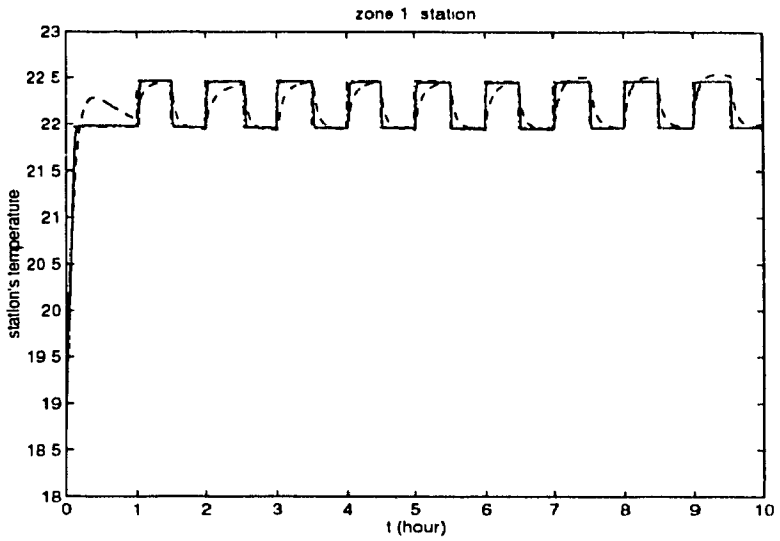

Figure 4.9: Station 2, comparison of the time-domain response of the NN based controller (solid line) and of the robust controller (dashed line). The dashdot line is the setpoint (almost matching the solid line).

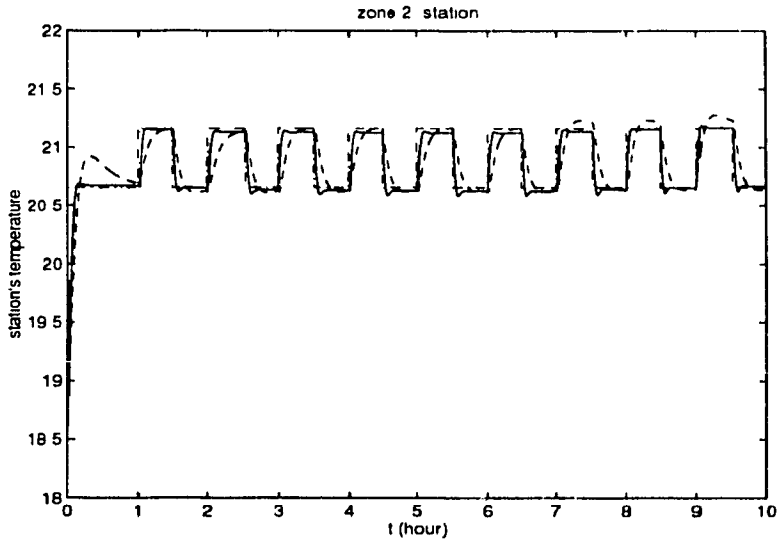

Figure 4.10: Station 1, comparison of the time-domain response of the NN based controller (solid line) and of the robust controller (dashed line). The dashdot line is the setpoint.

4.4.3 Occupied/Unoccupied Setpoint Control

In order to save energy in buildings, especially commercial ones, the temperature of zones and the boiler should be set back several degrees when the building is considered unoccupied. Such a change in the setpoint of control system usually requires design of more than one robust controller. Some approaches for designing such controllers exist such as preview control [42]. In such control schemes, there is a supervisory controller which adjust the gains of the main controller in order to avoid the violation of control system constraints and/or to keep the regulator gain values as close to optimal value as possible. Such adjustment is done based on predicting future conditions or measuring the present condition.

In the control problem at hand the neural network based controller performs both regulator and supervisory controller tasks. So, the gain values satisfy the system constraints for the conditions for which the neural networks are trained. It is likely that having real condition at neural networks training, which can be interpreted as regulator design, including constraint satisfaction and probable closeness of regulator gain values to their optimal values arises the hope that the performance of neural network based controller could be superior to that of other approaches. Figures 4.11-4.13 show the response of neural network based controller to occupied/unoccupied reference input changes. Note that zone 1 and zone 2 temperatures are set forward by 7 ^{o}C in about half-hour.

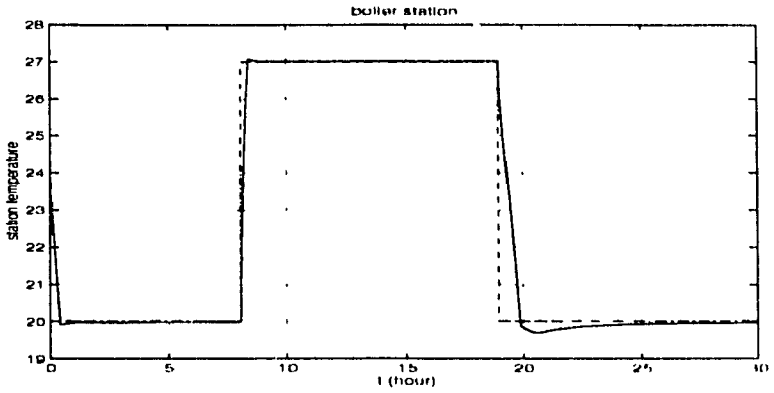

Figure 4.11: Station 1. Occupied/unoccupied temperature control. The solid line is the response of the neural network based controller and the dash-dot line is the reference input.

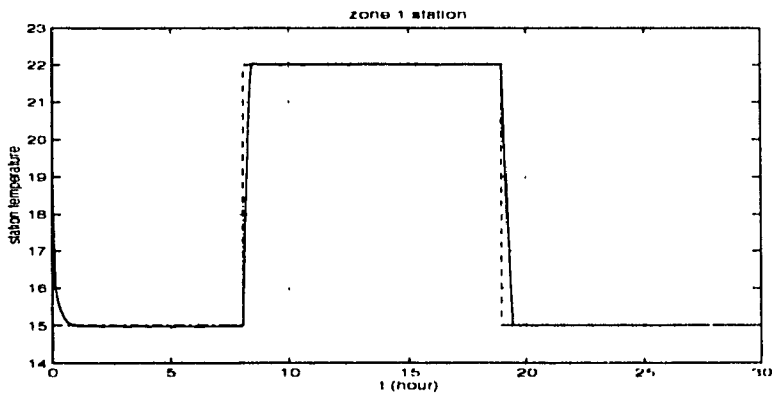

Figure 4.12: Station 2. Occupied/unoccupied temperature control. The solid line is the response of the neural network based controller and the dash-dot line is the reference input.

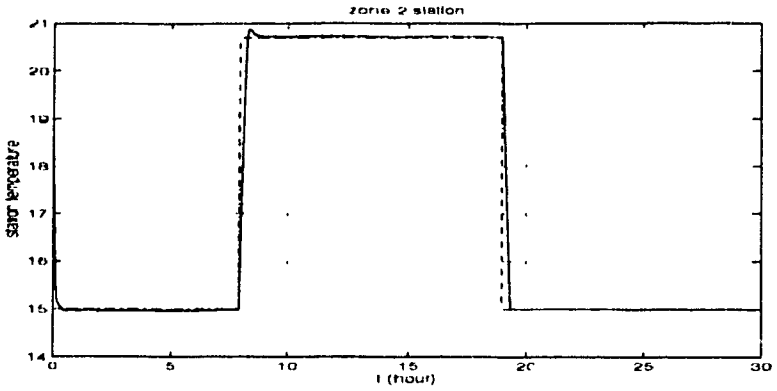

Figure 4.13: Station 3. Occupied/unoccupied temperature control. The solid line is the response of the neural network based controller and the dash-dot line is the reference input.

4.4.4 Disturbance Rejection

Disturbance rejection in a building is as important as occupant setpoint changes of the point of view of comfort. Since the disturbances acting on the control system are not strictly included in the training scheme for neural networks, there is no knowledge of how well the disturbance rejection of neural network based controller is. In order to study the effect of realistic disturbances on the neural network based controller, we consider a normal outdoor temperature of a winter's day in Montreal (Fig. 4.14).

Figures 4.16, 4.18, and 4.20 depict the response of the neural network based controller while having the setpoints at some fixed values, but with disturbances acting on the system. The responses show that the neural network based controller provides good disturbance rejection. Figures 4.15, 4.17, and 4.19 show the response of the robust controller under the same condition as the neural network based controller is.

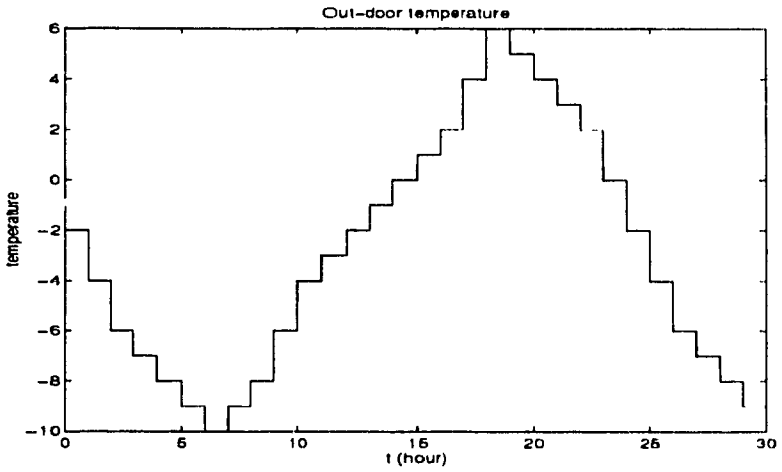

Figure 4.14: A normal winter day temperature in Montreal.

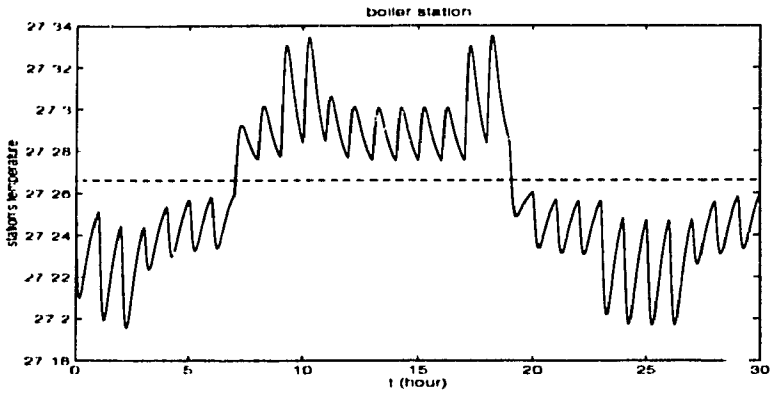

Figure 4.15: Station 1. The response of a robust controller (the solid line) to the disturbance applied to the system. The dashed line is the reference input value.

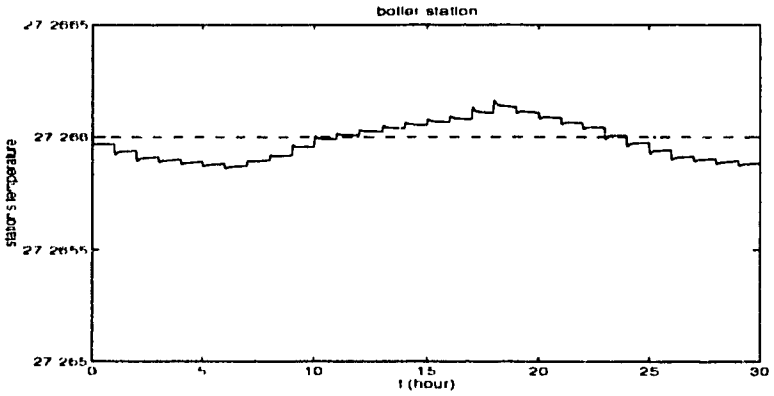

Figure 4.16: . Station 1. The response of the neural network based controller (the solid line) to the disturbance applied to the system. The dash-dot line is the reference input value.

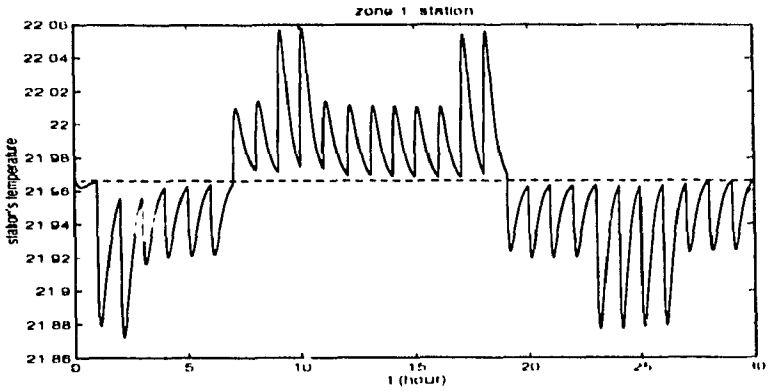

Figure 4.17: Station 2. The response of a robust controller (the solid line) to the disturbance applied to the system. The dashed line is the reference input value.

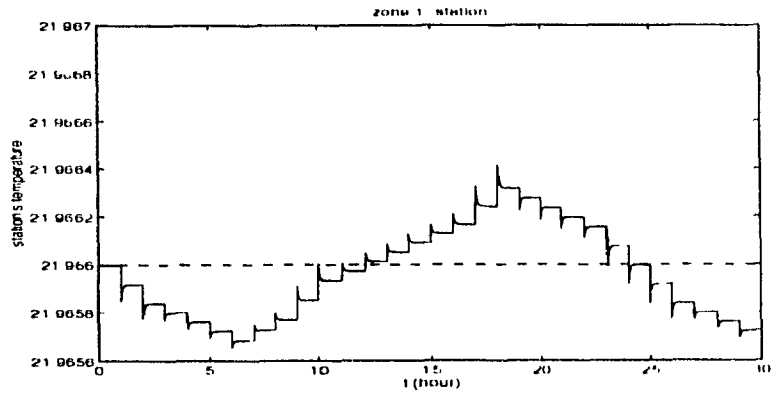

Figure 4.18: Station 2. The response of the neural network based controller (the solid line) to the disturbance applied to the system. The dash-dot line is the reference input value.

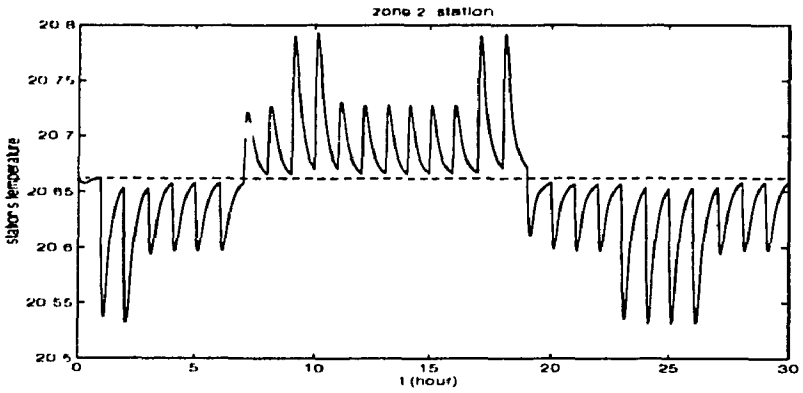

Figure 4.19: Station 3. The response of a robust controller (the solid line) to the disturbance applied to the system. The dashed line is the reference input value.

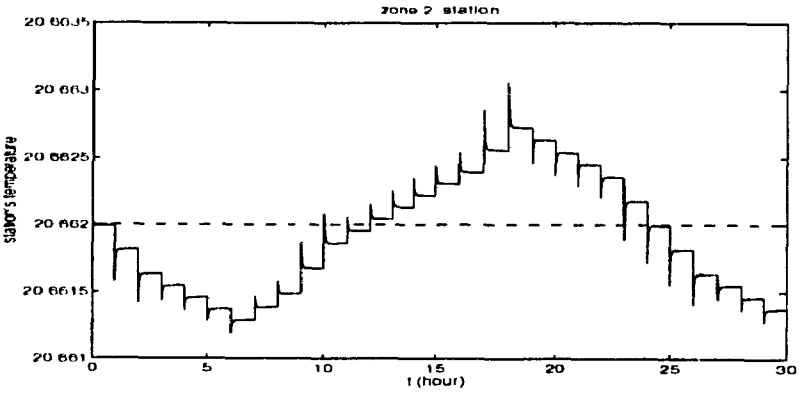

Figure 4.20: Station 3. The response of the neural network based controller (the solid line) to the disturbance applied to the system. The dash-dot line is the reference input value.

4.4.5 Neural Network Recall

This part of the study covers two alternatives to on-line neural network based control; (i) Neural network recall with weight update and (ii) Feedback gain update.

(i) In the normal recall mode of operation of neural networks, it is assumed that the weight matrices within the neural network carry complete information required for controlling the system. This also means that a static mapping from the neural network inputs to the feedback gains can be found such that the performance of the controlled system is close to that of the on-line neural network based controlled system. Attempts for finding this static mapping failed.

The recall mode which successfully controlled the heating system was recalled with weight updating. For this kind of recall the changes in the weight matrices of the neural network, when under training, are saved. The weight matrices and their changes, i.e. time varying internal parameters for the neural network, are employed during the recall operation. In this scheme the time variation of the neural network internal parameters are not as fast as the variation of the internal parameters for the online controller.

(ii) In feedback gain updating the controller gains are updated based on neural network outputs while the network is under training. This test requires a lower gain update rate than that used in the neural network based control.

Although, the following figures show that the system is performing well the cost of operation is greater than that of the controller with neural networks while under training.

Control with NNs in Recall Mode

Figures 4.21-4.23 show the response of the system when the neural networks are in recall mode. The weight matrices of the neural networks are updated from a lookup

table every half an hour. The look-up table is made by saving neural network internal weight matrices obtained during neural network training. The conditions of this control system are the same as those of the on-line trained controller, but the neural networks internal parameters are changed more slowly.

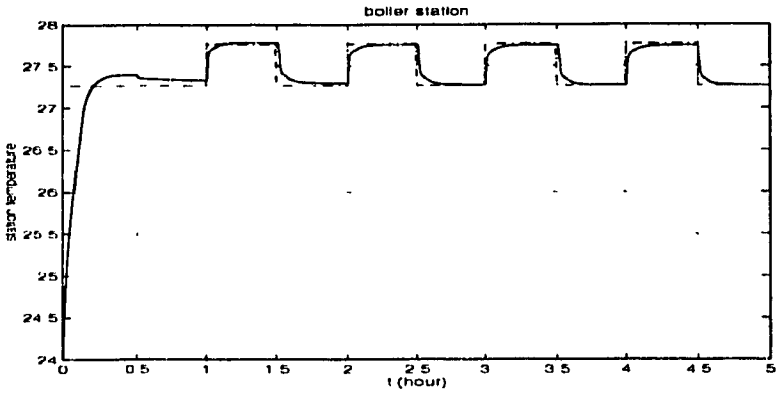

Figure 4.21: Station 1. Neural network based control with weight updating. The solid line is the response of the neural network based controller to the reference input variations (dash-dot line)

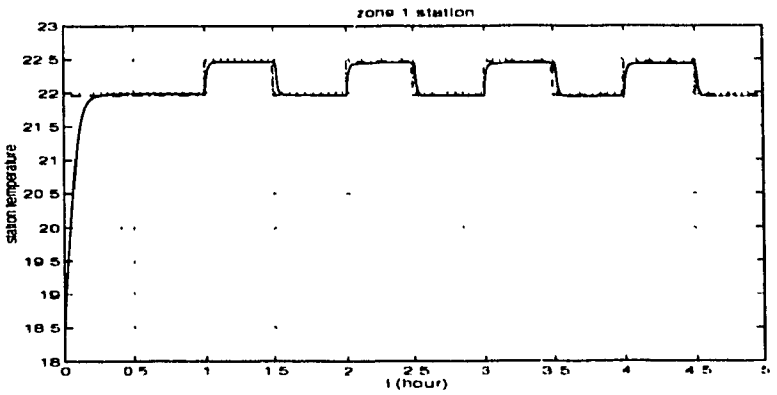

Figure 4.22: Station 2. Neural network based control with weight updating. The solid line is the response of the neural network based controller to the reference input variations (dash-dot line)

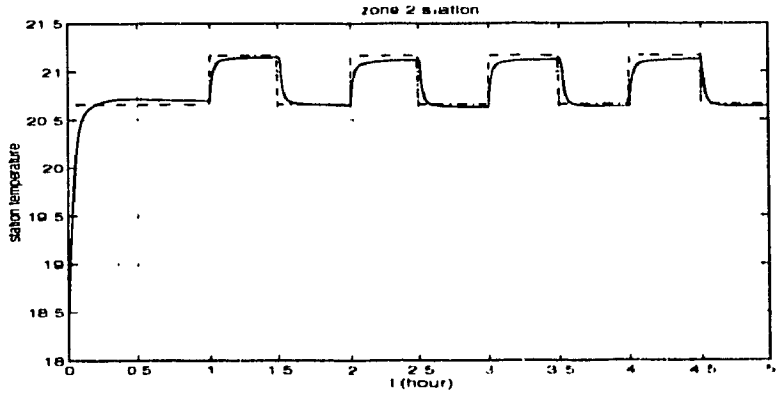

Figure 4.23: Station 3. Neural network based control with weight updating. The solid line is the response of the neural network based controller to the reference input variations (dash-dot line)

Gain Updating Control

Gain update performed by a lookup table. The look-up table is created by measuring the neural network responses when under training on-line. In the following simulations, the controller gains are updated every 0.1 hour, as if a trained neural network was in controlled system. This simulation actually presents the performance of the controlled system when the gain update rate is reduced. Although the cost of operation is not as low as other applications its implementation is easier than that of neural network based controller. Obviously any unpredicted condition in control system or any fast change of variables in the system would have to be taken care of by robustness of the controller. Figures 4.24-4.26 depict the response of the controlled system with gain update.

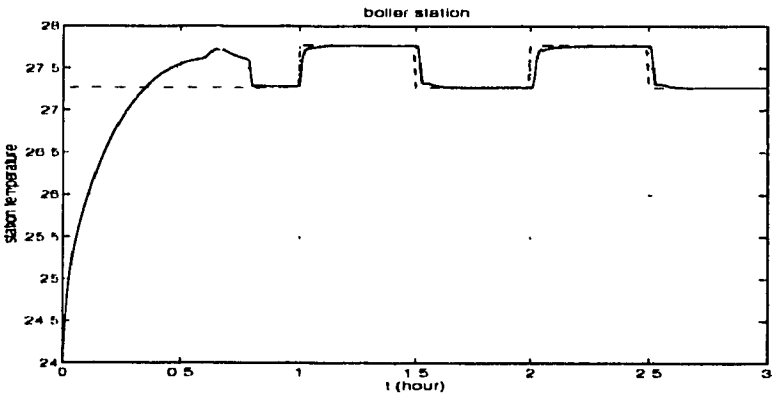

Figure 4.24: Station 1. Feedback gain updating.

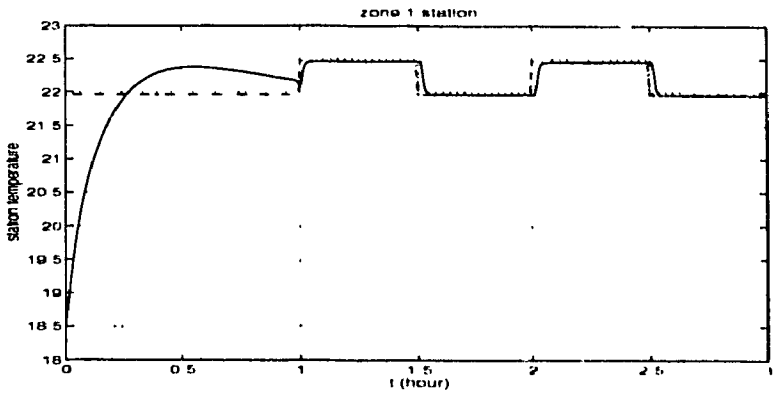

Figure 4.25: Station 2. Feedback gain updating.

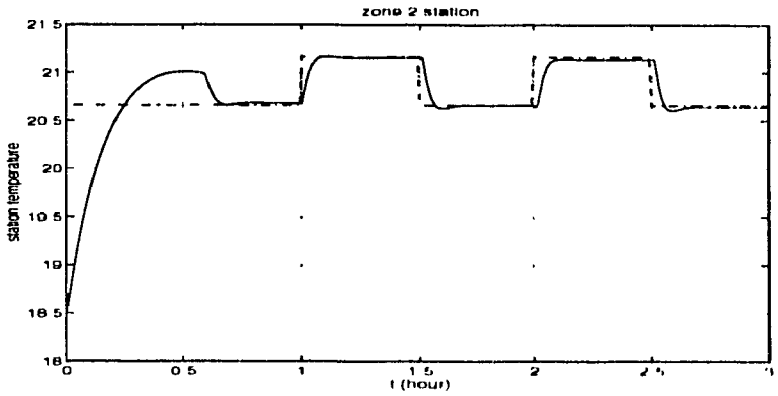

Figure 4.26: Station 3. Feedback gain updating.

Bibliography

[1] Saboksayr S. H., Patel R. V., Zaheer-uddin M., Energy-Efficient Operation of HVAC Systems using Neural Network based Decentralized Controllers, Proceedings of the American Control Conference, seattle, WA, June 1995. pp.4321-4325

[2] Nesler C. G., Adaptive Control of Thermal Processes in Buildings, IEEE Control Systems Magazine, Aug. 1986, pp. 9-13

[3] Wallenborg A. O., A New Self-Tuning controller for HVAC Systems, ASHRAE Transactions, vol. 97, Part 1, 1991, pp. 19-25

[4] Zaheeruddin M., and Patel R. V. The Design and Simulation of a Sub-Optimal Controller for Space Heating, ASHRAE Transactions, vol. 99, part 1, 1993, pp. 554-564

[5] He X., Liu S., Asad H., Multivariable Feedback Design for Regulating Vapor Compression Cycles, Proceedings of the American Control Conference, seattle, WA, June 1995. pp. 4331-4335.

[6] Zaheeruddin M., Patel R. V., and Al-Assadi S. A. K., Decentralized Control of a Variable Air Volume System, Proceedings of the American Control Conference, Baltimore, Maryland, June 1994, pp. 3050-3054

[7] Curtice P.S., Kreider J. F., and Brandemuehl M. J., Local and Global Control of Commercial Building HVAC Systems Using Artificial Neural Netowrks. Proceedings of the American Control Conference. June 1994. pp. 3029-3034.

[8] Curtice P.S., Kreider J. F., and Brandemuehl M. J., Adaptive Control of HVAC Processes Using Predictive Neural Networks, ASHRAE Transactions, vol.99, Part1, 1993, pp.496-504

[9] Zaheer-uddin M., Patel R. V., Al-Assadi S., Design of Decentralized Robust Controllers for Multizone Space Heating Systems, IEEE Transactions on Control Systems Technology, vol. 1, No. 4, December 1993.

[10] Davison E. J., The robust control of a servomechanism problem for linear time-invariant multivariable systems, IEEE Transaction on Automatic Control, vol. AC-21, pp. 25-34, 1976.

[11] Davison E. J. and Goldberg A., The robust control of a general servomechanism problem: the servo compensator, Automatica, vol. 11, pp. 461-471, 1975.

[12] Desoer C. A. and Wang Y. T., linear time-invariant robust servomechanism problem: A self-contained exposition, in Control and Dynamic Systems, C. T. Leondes, editor, vol. 16, 1980.

[13] Francis B., Sebakhy O. A., and Wonham W. M., Synthesis of multivariable regulator: the internal model principle, Appl. Math. Optimiz., vol. 1, pp. 64-86, 1974.

[14] Patel R. V. and Munro N., Multivariable System Theory and Design, Pergamon Press, 1981.

[15] Davison E. J. and Solomon A., Partial Decentralized Temperature Control of Multizone Buildings, proceedings of IEEE Conference on Decision and Control, San Antonio, TX, 1983, pp. 10-16.

[16] Davison E. J. The Robust Decentralized Control of a General Servomechanism Peoblem, IEEE Transactions on Automatic Control, vol. AC-21, N0. 1, Feb. 1976, pp.14-24.

[17] Warwick K., Irwin G. W., and Hunt K. J., Neural Networks for Control and Systems, Peter Peregrinus Ltd. 1992.

[18] Jacek M. Zurada. Introduction to Artificial Neural Systems, West Publishing Company

[19] Widrow B., Lehr M. A., 30 Years of Adaptive Neural Networks: Perceptron, Madaline, and backpropagation, Proceedings of the IEEE, vol. 78, No. 9, pp. 1415-1442, Sep. 1990.

[20] Rumelhart D. E. and McClelland J. L., Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol.1: Foundations, MIT Press. 1986.

[21] Bruce A. D., Canning A., Forest B., Gardner E., Walace D. J., Learning and memory properties in fully connected networks, Neural Networks for computing, NewYork, American Institute of Physics, 1986, pp. 65-70.

[22] Freeman J. A., Skapura D. M., Neural Networks: Algorithms, Applications, and Programming Techniques. Addison-Wesley Publication Company, 1992

[23] Lippmann R. P , An introduction to computing with neural nets, IEEE ASSP Magazine, vol. 4, April 1987, pp. 4-22.

[24] Cybenko G., Approximation by Superpositions of a Sigmoidal Function, Mathematics of Control, Signals and Systems, vol. 2, Oct. 1989, pp. 303-314.

[25] Werbos P. I., Beyond regression: New Tools for Prediction and Analysis in the Behavior Sciences, Ph.D. thesis, Harvard University, Cambridge MA. 1974.

[26] Piche X. W., Steepest Descent Algorithms for Neural Network Controllers and filters", IEEE Transactions on Neural Networks, vol.5, No.2, March 1994, pp. 198-212.

[27] Barnard E., Optimization for Training Neural Networks, IEEE Transactions on Neural Networks, vol.3, No. 2, March 1992, pp. 232-240.

[28] Cater J. P., Successfully using Peak learning rates of 10 (and greater) in backpropagation networks with the heuristic learning algorithm, Proc. Conf. Neural Networks san Diego, CA, July 1987, pp. II-645-II-651.

[29] Vogl T. P., Mangis J. K., Zigler A. K., Zink W. T., and Alkon D. L., Accelerating the convergence of the backpropagation method, Bio Cybern., vol. 59, pp. 257-264, Sept. 1988.

[30] Jacobs R. A., Increased rates of convergence through learning rater adaptation, Neural Networks, vol. 1, No. 4, pp. 295-308, 1988.

[31] Hsin H. C., Sun M., and Sclabassi R. J., An Adaptive Training Algorithm for Back-Propagation Neural Networks, IEEE Transactions on Systems, Man, and Cybernetics, vol. 25, No. 3, March 1995, pp. 512-514

[32] Widrow B. and Stearns S. D., Adaptive Signal Processing, Englewood Cliffs, NJ: Prentice-Hall, 1985.

[33] Parker D. B., Optimization algorithms for adaptive neural networks: second order backpropagation, second order direct propagation, and second order Hebbian learning, Proc. 1st IEEE International Conf. on Neural Networks, vol. 2, San Diego, CA, June 1987, pp. 593-600.

[34] Kramer A. and Sangiovanni-Vincentelli A., Efficient Parallel learning algorithms for neural networks, in Advances in Neural Information Processing Systems I, Morgan Kaufmann, 1989.

[35] Jin L., Nikiforuk P. N., and Gupta M. M., Fast Neural Learning and Control of Discrete-Time Nonlinear Systems, IEEE Transactions on Systems, Man, and Cybernetics, vol. 25, No. 3, March 1995, pp. 478-488

[36] R. Hecht-Nielsen, Theory of the backpropagation neural network, Proceedings of International Joint Conference on Neural Networks, vol. I, (Washington D.C.), 1989, pp. 593-605.

[37] Narendra K. S. and Parthasarathy K., Identification and Control of Dynamic Systems Using Neural Networks, IEEE Transactions on Neural Networks, No. 1, 1990, pp. 4-27.

[38] Narendra K. S., Parthasarathy K., Gradient Methods for the Optimization of Dynamical Systems Containing Neural Networks, IEEE Transactions on neural networks, vol. 2, No. 2, March 1991, pp. 252-262.

[39] Alexander Graham, Kronecker Products and matrix calculus with applications, John Wiley Sons, 1981.

[40] Adby P. R., Dempster M. A. H., Introduction to Optimization Methods, Chapman and Hall, 1982.

[41] Saerens, M. and Soquet, A., A neural controller, 1st IEE conference on Neural Networks, London, Eng., Oct. 1989, pp. 211-215.

[42] Zaheer-uddin M., Al-Assadi S. A. K., Patel R. V., Decentralized Preview control for Multiple Disturbance Rejection in HVAC systems, CEP, vol. 2 (6), 1994, pp. 989-1000.

Appendix A

Matrices of the Linearized Models of the MZSH System

A.1 Linearized Model of the MZSH System

The linearized model matrices of the plant which is defined in Chapter 2 are as follows.

$$A = \begin{bmatrix} -19.525 & 7.988 & 0 & 0 & 7.988 & 0 & 0 \\ 28.361 & -29.759 & 1.134 & 0 & 0 & 0 & 0 \\ 0 & 1.134 & -9.529 & -8.130 & 0 & 0 & 0 \\ 0 & 0 & 3.635 & -5.054 & 0 & 0 & 0.2364 \\ 28.361 & 0 & 0 & 0 & -29.759 & 1.134 & 0 \\ 0 & 0 & 0 & 0.2659 & 0 & 4.090 & -5.685 \end{bmatrix} \quad (A.1)$$

$$B = \begin{bmatrix} 0 & 0 & 80.698 & 0 & 0 & 137.428 & 0 \\ 0 & 0 & 0 & -92.442 & 0 & 0.2642 & 0 \\ -160.130 & 0 & 0 & 133.284 & 0 & 0.2642 & 0 \\ 71.586 & 0 & 0 & 0 & 0 & 0 & 1.1818 \\ 0 & 0 & 0 & 0 & -96.586 & 0.2642 & 0 \\ 0 & -148.033 & 0 & 0 & 137.428 & 0.2642 & 0 \\ 0 & 66.189 & 0 & 0 & 0 & 0 & 1.3296 \end{bmatrix} \quad \text{(A.2)}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(A.3)}$$

$$E = \begin{bmatrix} 1.774 & 0.2642 & 0.2642 & 0 & 0.2642 & 0.2642 & 0 \\ 0 & 0 & 0 & 1.1818 & 0 & 0 & 1.3296 \end{bmatrix} \quad \text{(A.4)}$$

A.2 Decentralized Linear Model of the MZSH System

The decentralized model matrices which can be obtained from the centralized model are:

$$A = \begin{bmatrix} -19.525 & 7.988 & 0 & 0 & 7.988 & 0 & 0 \\ 28.361 & -29.759 & 1.134 & 0 & 0 & 0 & 0 \\ 0 & 1.134 & -9.529 & -8.130 & 0 & 0 & 0 \\ 0 & 0 & 3.635 & -5.054 & 0 & 0 & 0.2364 \\ 28.361 & 0 & 0 & 0 & -29.759 & 1.134 & 0 \\ 0 & 0 & 0 & 0.2659 & 0 & 4.090 & -5.685 \end{bmatrix} \quad (A.5)$$

$$B_1 = \begin{bmatrix} 80.698 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 0 & 0 \\ 0 & -92.442 \\ -160.103 & 133.284 \\ 71.586 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \quad B_3 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & -96.586 \\ -148.033 & 137.428 \\ 66.1890 & \end{bmatrix} \quad (A.6)$$

$$C_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad (A.7)$$

$$C_2 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix},$$

$$C_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

A.3 Linear Model of the Augmented System

The matrices determining the augmented system for the robust servomechanism problem, centralized case, are as follows:

$$A = \begin{bmatrix} A & 0 \\ \Theta C & \Omega \end{bmatrix} \tag{A.8}$$

$$= \begin{bmatrix} -19.525 & 7.988 & 0 & 0 & 7.988 & 0 & 0 & 0 & 0 & 0 \\ 28.361 & -29.759 & 1.134 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1.134 & -9.529 & -8.130 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3.635 & -5.054 & 0 & 0 & 0.2364 & 0 & 0 & 0 \\ 28.361 & 0 & 0 & 0 & -29.759 & 1.134 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.2659 & 0 & 4.090 & -5.685 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} B \\ 0 \end{bmatrix} \tag{A.9}$$

$$= \begin{bmatrix} 0 & 0 & 80.698 & 0 & 0 & 137.428 & 0 \\ 0 & 0 & 0 & -92.442 & 0 & 0.2642 & 0 \\ -160.130 & 0 & 0 & 133.284 & 0 & 0.2642 & 0 \\ 71.586 & 0 & 0 & 0 & 0 & 0 & 1.1818 \\ 0 & 0 & 0 & 0 & -96.586 & 0.2642 & 0 \\ 0 & -148.033 & 0 & 0 & 137.428 & 0.2642 & 0 \\ 0 & 66.189 & 0 & 0 & 0 & 0 & 1.3296 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\bar{C} = \begin{bmatrix} C & 0 \\ 0 I_{pq} & \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.10}$$

$$\bar{E} = \begin{bmatrix} E \\ 0 \end{bmatrix} = \begin{bmatrix} 1.774 & 0.2642 & 0.2642 & 0 & 0.2642 & 0.2642 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.1818 & 0 & 0 & 1.3296 & 0 & 0 & 0 \end{bmatrix}' \tag{A.11}$$

$$\bar{F} = \begin{bmatrix} 0 \\ -\Theta \end{bmatrix} = \begin{bmatrix} 0_{7\times 3} \\ -I_{3\times 3} \end{bmatrix} \tag{A.12}$$

A.4 Decentralized Linear Model of the Augmented System

The matrices required for the decentralized form of linear robust servomechanism problem are as follows:

$$\hat{A} = \begin{bmatrix} -19.525 & 7.988 & 0 & 0 & 7.988 & 0 & 0 & 0 & 0 & 0 \\ 28.361 & -29.759 & 1.134 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1.134 & -9.529 & -8.130 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3.635 & -5.054 & 0 & 0 & 0.2364 & 0 & 0 & 0 \\ 28.361 & 0 & 0 & 0 & -29.759 & 1.134 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.2659 & 0 & 4.090 & -5.685 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \quad \text{(A.13)}$$

$$\hat{B}_1 = \begin{bmatrix} 80.698 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \hat{B}_2 = \begin{bmatrix} 0 & 0 \\ 0 & -92.442 \\ -160.103 & 133.284 \\ 71.586 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \quad \hat{B}_3 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & -96.586 \\ -148.033 & 137.428 \\ 66.1890 & \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \quad \text{(A.14)}$$

$$\bar{C}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}, \quad \text{(A.15)}$$

$$\bar{C}_2 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix},$$

$$\bar{C}_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$E = \begin{bmatrix} 1.774 & 0.2642 & 0.2642 & 0 & 0.2642 & 0.2642 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.1818 & 0 & 0 & 1.3296 & 0 & 0 & 0 \end{bmatrix} \quad \text{(A.16)}$$

$$\hat{F} = \begin{bmatrix} 0 \\ -\Theta \end{bmatrix} = \begin{bmatrix} 0_{7\times3} \\ -I_{3\times3} \end{bmatrix} \quad \text{(A.17)}$$

Appendix B

The Equations of the Gradient for the Cost Function

This appendix presents the gradient equations which were used in the negative gradient optimization problem in this thesis. For minimizing the cost function dynamically, a window in the time-domain can be considered in which the variation of the cost function with respect to each feedback gain can be found. For example, for the interval between t_1 and t_2 we would get,

$$\frac{\partial J}{\partial k_1} = \int_{t_1}^{t_2} \{\sum_{i=1}^{3} 2\, e_i(t)\, \frac{\partial e_i(t)}{\partial u_1(t)}\, e_2(t) + r\, u_1(t)\, e_2(t) + r\, [u_2(t)\, k_3 + u_5(t)\, k_9]\, \frac{\partial e_3(t)}{\partial u_1(t)} + r\, u_4(t)\, k_7\, \frac{\partial e_2(t)}{\partial u_1(t)}\, e_2(t)\} dt\, e_2(t)\} dt \tag{B.1}$$

(B.2)

$$\frac{\partial J}{\partial k_2} = \int_{t_1}^{t_2} \{\sum_{i=1}^{3} 2\, e_i(t)\, \frac{\partial e_i(t)}{\partial u_1(t)}\, \xi_2(t) + r\, u_1(t)\, \xi_2(t) + r\, [u_2(t)\, k_3 + u_5(t)\, k_9]\, \frac{\partial e_3(t)}{\partial u_1(t)}\, \xi_2(t) + r\, u_3(t)\, k_5\, \frac{\partial e_1(t)}{\partial u_1(t)}\, \xi_2(t)\} dt$$

(B.3)

$$\frac{\partial J}{\partial k_3} = \int_{t_1}^{t_2} \{\sum_{i=1}^{3} 2\, e_i(t)\, \frac{\partial e_i(t)}{\partial u_2(t)}\, e_3(t) + r\, [u_1(t)\, k_1 + u_4(t)\, k_7]\, \frac{\partial e_2(t)}{\partial u_2(t)}\, e_3(t) + r\, u_2(t)\, e_3(t) + r\, u_3(t)\, k_5\, \frac{\partial e_1(t)}{\partial u_2(t)} + r\, u_5(t)\, k_9\, \frac{\partial e_3(t)}{\partial u_2(t)}\, e_3(t)\} dt$$

(B.4)

$$\frac{\partial J}{\partial k_4} = \int_{t_1}^{t_2} \{\sum_{i=1}^{3} 2\, e_i(t)\, \frac{\partial e_i(t)}{\partial u_2(t)}\, \xi_3(t) + r\, [u_1(t)\, k_1 + u_4(t)\, k_7]\, \frac{\partial e_2(t)}{\partial u_2(t)}\, \xi_3(t) + r\, u_2(t)\, \xi_3(t) + r\, u_3(t)\, k_5\, \frac{\partial e_1(t)}{\partial u_2(t)} + r\, u_5(t)\, k_9\, \frac{\partial e_3(t)}{\partial u_2(t)}\, \xi_3(t)\} dt$$

(B.5)

$$\frac{\partial J}{\partial k_5} = \int_{t_1}^{t_2} \{\sum_{i=1}^{3} 2\, e_i(t)\, \frac{\partial e_i(t)}{\partial u_3(t)}\, e_1(t) + 2\, r\, [u_1(t)\, k_1 + u_4(t)\, k_7]\, \frac{\partial e_2(t)}{\partial u_3(t)}\, e_1(t) + r\, [u_5(t)\, k_9 + u_4(t)\, k_7]\, \frac{\partial e_3(t)}{\partial u_3(t)}\, e_1(t) + r\, u_3(t)\, e_1(t)\} dt$$

(B.6)

$$\frac{\partial J}{\partial k_6} = \int_{t_1}^{t_2} \{\sum_{i=1}^{3} 2\, e_i(t)\, \frac{\partial e_i(t)}{\partial u_3(t)}\, \xi_1(t) + 2\, r\, [u_1(t)\, k_1 + u_4(t)\, k_7]\, \frac{\partial e_2(t)}{\partial u_3(t)}\, \xi_1(t) + r\, [u_5(t)\, k_9 + u_4(t)\, k_7]\, \frac{\partial e_3(t)}{\partial u_3(t)}\, \xi_1(t) + r\, u_3(t)\, \xi_1(t)\} dt$$

(B.7)

$$\frac{\partial J}{\partial k_7} = \int_{t_1}^{t_2} \{\sum_{i=1}^{3} 2\, e_i(t)\, \frac{\partial e_i(t)}{\partial u_4(t)}\, e_2(t) + 2\, r\, u_1(t)\, k_1\, \frac{\partial e_1(t)}{\partial u_4(t)}\, e_2(t) + 2\, r\, [u_2(t)\, k_3 + u_5(t)\, k_9]\, \frac{\partial e_3(t)}{\partial u_4(t)}\, e_2(t) + r\, u_3(t)\, k_5\, \frac{\partial e_1(t)}{\partial u_4(t)}\, e_2(t) + 2\, r\, u_4(t)\, e_2(t)\} dt$$

(B.8)

$$\frac{\partial J}{\partial k_8} = \int_{t_1}^{t_2} \{\sum_{i=1}^{3} 2\, e_i(t)\, \frac{\partial e_i(t)}{\partial u_4(t)}\, \xi_2(t) + 2\, r\, u_1(t)\, k_1\, \frac{\partial e_1(t)}{\partial u_4(t)}\, \xi_2(t) + 2\, r\, [u_2(t)\, k_3 + u_5(t)\, k_9]\, \frac{\partial e_3(t)}{\partial u_4(t)}\, \xi_2(t) + r\, u_3(t)\, k_5\, \frac{\partial e_1(t)}{\partial u_4(t)}\, \xi_2(t) + 2\, r\, u_4(t)\, e_2(t)\} dt$$

(B.9)

$$\frac{\partial J}{\partial k_9} = \int_{t_1}^{t_2} \{\sum_{i=1}^{3} 2\, e_i(t)\, \frac{\partial e_i(t)}{\partial u_5(t)}\, e_3(t) + 2\, r\, [u_1(t)\, k_1 + u_4(t)\, k_7]\, \frac{\partial e_2(t)}{\partial u_5(t)}\, e_3(t) + 2\, r\, u_2(t)\, k_3\, \frac{\partial e_3(t)}{\partial u_5(t)}\, e_3(t) + 2\, r\, u_3\, k_5\, \frac{\partial e_1(t)}{\partial u_5(t)}\, e_3(t) + 2\, r\, u_5(t)\, e_3(t)\} dt$$

(B.10)

$$\frac{\partial J}{\partial k_{10}} = \int_{t_1}^{t_2} \{\sum_{i=1}^{3} 2\, e_i(t)\, \frac{\partial e_i(t)}{\partial u_5(t)}\, \xi_3(t) + 2\, r\, [u_1(t)\, k_1 + u_4(t)\, k_7]\, \frac{\partial e_2(t)}{\partial u_5(t)}\, \xi_3(t)$$
$$+ 2\, r\, u_2(t)\, k_3\, \frac{\partial e_3(t)}{\partial u_5(t)}\, \xi_3(t) + 2\, r\, u_3\, k_5\, \frac{\partial e_1(t)}{\partial u_5(t)}\, \xi_3(t) + 2\, r\, u_5(t)\, \xi_3(t)\} dt$$

For minimizing the decentralized costs, J_1, J_2 and J_3 defined in Chapter 4, similar equations can be found which neglect the relationship between the output errors and the control inputs which do not belong to the same station of the multizone space heating system. Another way of finding the cost function variation is suggested in Appencix C.

Appendix C

Another Method of Computing the Cost Function Gradient

In order to find the variation of the cost function with respect to the variation of every controller gain, we consider Figure C.1 as the model of the control system. Consider the governing equations of the controller, 4.3 and 4.4. Any variation in an

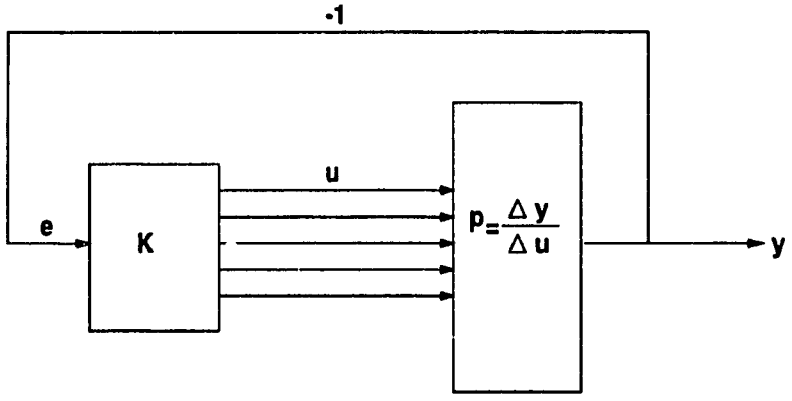

Figure C.1: The model for finding the gradient of the cost function

element of K will cause some change in the corresponding $\nu(t)$, i.e.

$$\Delta k_1 \rightarrow \Delta\nu_1(t) = \Delta k_1\, e_2(t) \tag{C.1}$$

$$\Delta k_2 \rightarrow \Delta\nu_1(t) = \Delta k_1\, \xi_2(t) \tag{C.2}$$

$$\Delta k_3 \rightarrow \Delta\nu_2(t) = \Delta k_3\, e_3(t) \tag{C.3}$$

$$\Delta k_4 \rightarrow \Delta\nu_2(t) = \Delta k_4\, \xi_3(t) \tag{C.4}$$

$$\Delta k_5 \rightarrow \Delta\nu_3(t) = \Delta k_5\, e_1(t) \tag{C.5}$$

$$\Delta k_6 \rightarrow \Delta\nu_3(t) = \Delta k_6\, \xi_1(t) \tag{C.6}$$

$$\Delta k_7 \rightarrow \Delta\nu_4(t) = \Delta k_6\, e_2(t) \tag{C.7}$$

$$\Delta k_8 \rightarrow \Delta\nu_4(t) = \Delta k_8\, \xi_2(t) \tag{C.8}$$

$$\Delta k_9 \rightarrow \Delta\nu_5(t) = \Delta k_9\, e_3(t) \tag{C.9}$$

$$\Delta k_{10} \rightarrow \Delta\nu_5(t) = \Delta k_{10}\, \xi_3(t) \tag{C.10}$$

So, any variation in K, i.e. ΔK, can be presented in terms of an appropriate $\Delta\nu$ at the input of the plant. For example the effect of Δk_1 can be found by using the model of Figure C.2. By assuming a known small value of Δk_1, we would get all $\frac{\Delta e_i}{\Delta k_1}$ for $1 \leq i \leq 3$ and all $\frac{\Delta \nu_i}{\Delta k_1}$ from the model, i.e., at a specific instant of time we would get the following:

$$\frac{\Delta u(t_1)}{\Delta k_1} = (I + KP)^{-1} \begin{bmatrix} e_2(t_1) \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{C.11}$$

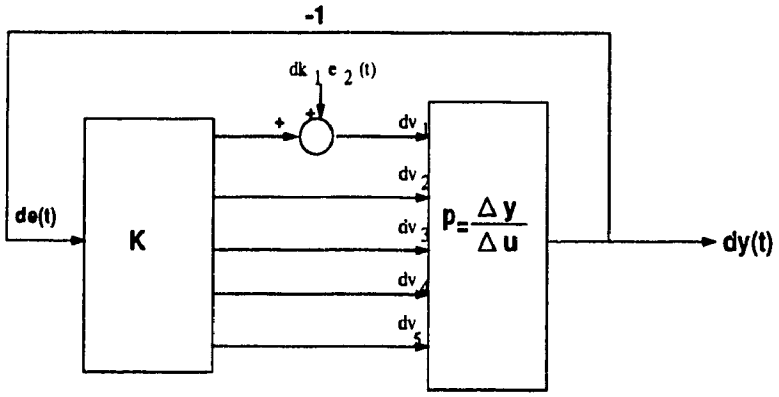

Figure C.2: Dtails of the model for finding the gradient of the cost function.

$$\frac{\triangle e(t_1)}{\triangle k_1} = -P(I+KP)^{-1}\begin{bmatrix} e_2(t_1) \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{C.12}$$

and

$$\frac{\triangle u(t_1)}{\triangle k_2} = (I+KP)^{-1}\begin{bmatrix} \xi_2(t_1) \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{C.13}$$

$$\frac{\triangle e(t_1)}{\triangle k_2} = -P(I+KP)^{-1}\begin{bmatrix} \xi_2(t_1) \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{C.14}$$

Similarly, such expressions can be found for other controller gains. If the gradient, and consiquently the neural network training, take place in discrete time, as is

done in this study, all the above equations can be found at the proper instants of time, and assumed to be constant during the interval. Thus, the variation of the cost function with respect to each controller gain variation can be found for the considered interval and the dynamic cost minimization can take place. If the period of gradient evaluation is short, the above equations can introduce a significant computational burden.