The Application of Adaptive Fuzzy Control to a Warm Water Process

Stephen E. McCormac Dip. E.E. B.Sc. (Eng)

Submitted for the qualification of Master of Engineering

Dublin City University

Supervisor : Dr. John V. Ringwood

School of Electronic Engineering

March, 1995

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Master of Engineering is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed: Stephen McCann ID No. 93700598

Candidate

Date : 24/7/95

Acknowledgements

I would like to acknowledge the considerable technical expertise, support and commitment of my supervisor Dr John V. Ringwood, who was a key figure in assisting and guiding me during this research. In addition, thank you to all the members of staff of Dublin City University who contributed towards the completion of this research.

From the Department of Control Systems and Electrical Engineering in the Dublin Institute of Technology, Kevin Street, a special word of thanks is due to my two external supervisors, Dr Richard Hayes and Mr Colm Murray, whose technical and organisational skills were essential for this project. A special word of thanks is due to Mr Ciaron Young and Mr Rainer Kiewitt who assisted me with the many time consuming tests on the plant. Thank you, as well, to the other members of staff who assisted me during the course of this research.

For their efforts in the proof reading of this thesis, thank you to Miss Gerbera Ottenberg, Mr Peter Demel, Mr Enda O'Sullivan and Mr Damon Berry.

I wish all the members of the Control Systems Group in Dublin City University and the Industrial Controls Centre in the Dublin Institute of Technology the very best for their future careers.

A special thank you is due to my parents, Anne and Stephen J. McCormac, for their support and patience which have served to make this research possible.

Finally, a word of thanks is due to Professor Dr. -Ing. Gerry Byrne, Head of the Mechanical Engineering Department in University College Dublin. Without his initial support and advice this project would not have been likely.

Dedication

Gerbera - es hat sich doch gelohnt, oder?!

CONTENTS

The Application of Adaptive Fuzzy Control to a Warm Water Process

Stephen E. McCormac Dip. E.E. B.Sc. (Eng)

This thesis presents research performed to investigate the current state-of-the-art in the field of adaptive fuzzy control. This research consists of two main parts, a detailed literature survey and the application of a chosen adaptive fuzzy control strategy to a non-linear coupled multi-variable plant.

As a direct result of the literature survey, the field of adaptive fuzzy control is categorised into three main classes, stand-alone adaptive fuzzy control, neural-fuzzy adaptive fuzzy control and hybrid adaptive fuzzy control. This research focuses on stand-alone adaptive fuzzy controller, with emphasis placed on those strategies that adapt on-line during plant control. Both direct and indirect adaptive fuzzy control paradigms are chosen for closer investigation through simulation.

The non-linear coupled multi-variable plant used in this research is a warm water process which consists of a process reaction tank with hot and cold inlets and an outlet. The outlet temperature and flow variables serve as controlled variables. After the design, development and construction of a hardware/software interface between the plant and a computer, three modelling strategies are applied to the plant - physical first principles, artificial neural networks and adaptive fuzzy modelling. During the modelling of the plant, an adaptive fuzzy modelling strategy, entitled supervised adaptive fuzzy modelling, is developed.

Based on simulation results, an indirect adaptive fuzzy controller is chosen as a basis for control of the real plant. This control paradigm is developed into a single step predictive fuzzy control strategy utilising the supervised adaptive fuzzy models in combination with a reference model to control the plant. Single and multi-variable control are achieved in simulation while single variable control is realised on the real plant.

The thesis concludes with further suggestions for research which include an algorithm for the extension of the prediction horizon of the single step predictive fuzzy controller, thus creating the multi-step predictive fuzzy controller.

Chapter 1 - *General Introduction to the Thesis*

This research is concerned with the investigation into methods for adaptive fuzzy control. This control paradigm is investigated utilising two methods, the first of which is a review and evaluation of the current theories regarding adaptive fuzzy control paradigms. This theoretical discussion based on a detailed literature survey is then augmented by the application of an adaptive fuzzy control method to a non-linear plant. The practical application is hereby divided into four sections :

- the design and construction of interface hardware and software between the plant and a computer,
- the development and evaluation of plant models,
- the design and simulation of two adaptive fuzzy controller strategies and one classical adaptive control method for the plant using the plant models for evaluation and
- the appraisal of an adaptive fuzzy control paradigm through its application to the real plant.

1.1. Overview of Chapter Structure

Chapter 1, serving as an introduction to the thesis, presents in *Section 1.2.* a synopsis of adaptive fuzzy control. This is followed by a general description of the non-linear plant used for the practical application of an adaptive fuzzy control strategy in *Section 1.3.* The final *Section 1.4.* gives a detailed description of the structure of this thesis.

1.2. Adaptive Fuzzy Control

1.2.1. Development

Since the early nineteen eighties the number of industrial applications of fuzzy logic has grown dramatically [1]. This novel mathematical theory, first conceived by Lofti Zadeh in 1965, has since developed into a powerful and effective engineering tool [2,3]. Fuzzy logic expands the concept of classical bivalent set membership to enable the partial or "fuzzy" membership of an object to a set, thus enabling the incorporation of a more human approach to set theory. As sets can represent human concepts such as *large*, *hot* or *far*, fuzzy logic gives the engineer a mathematical

framework for the representation of expert human knowledge which is often intrinsically in a *fuzzy* linguistic form [1,2,3].

The main area of application for fuzzy logic is *process control* [1,2,3]. Five reasons often cited for the advocacy of the use of fuzzy logic for process control are [4]:

- shorter "time to market" for products utilising fuzzy logic,
- more robust control characteristics,
- control of processes that could not previously be controlled using conventional control strategies,
- no necessity for a control theory expert and
- a mathematical model of the system to be controlled is not required.

Through the use of fuzzy logic, the expert knowledge of a plant operator can be transformed into a control algorithm. This algorithmic representation of human expert knowledge can then be utilised for automated control of the plant in question. Many successful implementations of this type of controller have been reported [5].

As the awareness of fuzzy control developed within the engineering community so too did the need for adaptive fuzzy control strategies [6]. This adaptive capability was to enable better closed loop control of the non-linear time varying plant over a wider range of operating points as compared to its non-adaptive fuzzy controller counterpart. Moreover, it was hoped that, as a fuzzy control algorithm can be linguistically interpreted, an adaptive fuzzy controller could *teach* a human operator an improved control policy for the plant [6].

1.2.2. Categorisation

The first step undertaken in this research was the categorisation of adaptive fuzzy control strategies by means of a literature survey. Resulting from this study, three classes of adaptive fuzzy controllers will be used:

- *Neural-Fuzzy Methods* - combinations of artificial neural networks and fuzzy algorithms,

- *Hybrid Adaptive Fuzzy Controllers* - combinations of fuzzy algorithms and classical control strategies based on dynamic systems theory, and

- *Stand Alone Adaptive Fuzzy Controllers* - only utilising fuzzy algorithms.

The *Neural-Fuzzy Methods* can, for example, convert a fuzzy algorithm to a neural network structure and then train the so formed neural network with measured data. This corresponds to the initialisation of a neural network with linguistic expert knowledge [7]. Another standard approach among the neural-fuzzy methods is to utilise the learning methods from the neural network field to adapt some aspect of a fuzzy logic algorithm [8]. Such algorithms can be applied to either emulate the control behaviour of an operator or to develop an inverse model of a plant for use as controller [9].

The *Hybrid Adaptive Fuzzy Controller* paradigms combine the linguistic capabilities of fuzzy logic and the deterministically proven classical controller designs. One common example of the hybrid adaptive fuzzy controller is a self-tuning PID controller which utilises a fuzzy algorithm to tune the gain parameters of a PID controller [10]. This example thus exploits the algorithmic representation of how a human operator would tune a PID controller by storing the expert knowledge in a fuzzy algorithm.

This thesis, however, concentrates on the Stand Alone Adaptive Fuzzy Controller. This type of adaptive fuzzy controller utilises some form of adaptation algorithm to adjust the parameters of a fuzzy controller. The form of the fuzzy controller is of secondary importance, the main characteristic is that no classical controller or artificial neural networks are incorporated in the design. One example of this method is the use of orthogonal least squares to adapt an inverse model of the plant to be controlled [11].

1.2.3. Adaptation

The adaptation of a fuzzy controller can be performed by either on-line or off-line methods:

- The *off-line* methods utilise a simulation model of the plant to be controlled to optimise their simulated closed loop control performance through adaptation. Thus in order to achieve good control performance an accurate simulation model of the plant is required. The mathematical models of many plants are, however,

simplifications which are often inaccurate. This inaccuracy can lead to poor control performance of an off-line optimised adaptive fuzzy controller.

- In contrast to the off-line adaptive fuzzy controllers, the *on-line* methods are capable of adapting to the real plant while controlling it. This ensures that the adaptation of the fuzzy controller will serve to only improve its closed loop control performance.

This research places more emphasis on the on-line adaptation of a fuzzy controller as this strategy clearly has more potential for good closed loop control performance than the off-line adaptive fuzzy control methods.

1.3. Brief Description of the Chosen Plant

As it has been claimed that fuzzy logic control is especially suited to the control of non-linear ill-defined systems, just such a plant was deemed to be best suited for a practical examination of the capabilities of adaptive fuzzy controllers [12]. Because the adaptive fuzzy controller can require considerable processing time, the plant dynamics needed to be slow to allow low sampling frequencies and thus more processing time for the controller algorithms. Hence, a *warm water process plant with slow dynamics* was chosen as the plant for this research. This process plant can be characterised by the following :

- *Construction* - a cylindrical tank with one hot inlet, one cold inlet and one outlet. The inlet flows are controllable, both inlet temperatures are non-controllable with only the hot inlet temperature being measurable. The outlet flow and temperature are both measurable and controllable.

- *Multivariable system* - with outlet flow and temperature serving as the controlled variables and the hot and cold inlets as inputs resulting in a two input - two output system.

- *Non-linear* - the plant has a well known non-linear outlet flow characteristic and unknown non-linear outlet temperature behaviour.

- *Slow dynamics* - allowing slower sampling rates and thus more processing time.

- *Industrial sensors and actuators* - all sensors and actuators are currently used in industry with all transducer signals transmitted over 4-20mA current loops.

Having decided on this warm water process for the practical investigation to be performed in this research, the next step was *to design and construct both hardware and software* to allow the interfacing of the warm water process to an IBM compatible PC computer. The necessary interface hardware is an instrumentation amplifier to condition the warm water process sensor signals for analogue to digital conversion. The constructed amplifier is a six channel two stage amplifier with integrated anti-aliasing filters. The interface software is an interrupt driven software structure written in the ANSI C programming language that provides the framework for both data acquisition from and closed loop control of the warm water process.

The interfacing of the plant to a computer provided the basis for the development of plant models and the design and testing of an adaptive fuzzy control strategy for the warm water process. An overview of the modelling and control of the warm water process is given in the next Section 1.3 of this chapter which describes the structure of thesis and the contents of each chapter.

1.4. Overview of the Thesis Structure

This thesis consists of seven Chapters all of which - excluding this general introductory *Chapter 1* - are detailed in the following description :

- *Chapter 2 - "Fuzzy Logic and Fuzzy Control"*, introduces the fuzzy logic controller and defines some basic terminology from the field of fuzzy control which is used throughout the thesis. A detailed description of the current stand alone adaptive fuzzy control strategies is included with some attention given to the issues of fuzzy modelling, fuzzy hardware and CASE tools for the development of fuzzy logic algorithms. The chapter concludes with a general summary and the choice of two stand alone adaptive fuzzy algorithms for investigation.

- *Chapter 3 - "Warm Water Process Hardware and Software"*, gives a detailed physical description of the warm water process. This description includes the geometry of the process and all associated sensors and actuators. This is followed by the specification of the analogue to digital and digital to analogue converter cards . A detailed description of the design, construction and testing of the instrumentation amplifiers for conditioning of the plants sensor signals is given. The chapter is concluded by a description

of the interrupt driven software structure used for closed loop control of and data acquisition from the warm water process.

- *Chapter 4 - "Warm Water Process Modelling"*, covers all aspects of three mathematical models of the warm water process that were developed for this research :

    - the *Physical Model* based on physical first principles,
    - the *Artificial Neural Network Model* and
    - the *Fuzzy Model.*

  This chapter commences with the description of the calibration of the inlet and outlet flow meters of the warm water process. This is followed by a description of the design, parameterisation and performance of the PI controllers that were utilised for the linearisation of the hot and cold inlet valves.

  The *physical model* of the warm water process, consisting of equations of the mass flow and thermal behaviour of the warm water process is described. The determination of their parameters, the simulation of the model in the MATLAB/SIMULINK environment and the evaluation of the model are detailed.

  The next plant model to be described is the *artificial neural network model* of the warm water process. The architecture and learning mechanism as well as the training data acquisition are all detailed. The modelling performance of this model is investigated using a separate set of test data taken from the warm water process.

  The last modelling strategy is that of *Supervised Adaptive Fuzzy Modelling* which is based on a standard fuzzy model type. This research contributes a straightforward but effective supervisory function to the fuzzy model, improving the overall modelling accuracy. A detailed description of the development of this on-line adaptive fuzzy modelling strategy using a linear first order system as an example is given. This is followed by a description of the development, the structure, learning mechanism and performance of

separate adaptive fuzzy models for the mass flow and thermal behaviour of the real plant.

- *Chapter 5 - "Controller Design"*, deals with the design and evaluation of two adaptive fuzzy control strategies and a deterministic adaptive control strategy:

    - *Self-Organising Control* - a direct adaptive fuzzy control method which adapts the rules of a fuzzy controller based on the current controller performance.

    - *Single Step Predictive Control* - an indirect adaptive control strategy which is based on the classical predictive controller. The supervised adaptive fuzzy models of the plant, developed in *Chapter 4*, are used to predict plant behaviour over a single step prediction horizon. Utilising a user defined reference model, a simple search algorithm chooses the controller output that best minimises a given cost function.

    - *Self-tuning PI control* - is developed from a self-tuning PID control strategy. This controller uses on-line RLS identification of a first order ARX model of the plant to be controlled to calculate the parameters of a PI controller so that a user defined dynamic response is fulfilled.

All three adaptive control paradigms are compared and contrasted. Based on these evaluations the Single Step Predictive Controller (SPFC) is chosen for control of the warm water process.

The key contribution of this research to the area of fuzzy control, is the extension of the prediction horizon of the single step predictive fuzzy controller, thus creating the *Multistep Predictive Fuzzy Controller* (MPFC). The suggested MPFC controller is summarised by the following points :

1. the utilisation of the supervised adaptive fuzzy model structure, developed in Chapter 4 of this thesis, to allow on-line adaptation of the fuzzy model,

2. the modification of the supervised adaptive fuzzy model structure and parameters to allow differentiation of the model outputs with respect to its inputs and

3. the application of a gradient descent algorithm to enable efficient calculation of the predicted plant outputs and corresponding manipulated variable values whilst minimising a user defined cost function over a multistep prediction horizon.

A full description of the proposed MPFC with the necessary mathematical derivations for this controller paradigm is given. A practical implementation of this multistep predictive fuzzy controller is not carried out as it is deemed to be beyond the scope of this thesis.

- *Chapter 6 - "Real Time Control of the Warm Water Process"*, presents the results of a series of tests performed on the real warm water process to evaluate the control performance of the Single Step Adaptive Predictive Controller.

- *Chapter 7 - "Further Research and Conclusions"*, is the final chapter of this thesis and assesses the research undertaken in this thesis, and offers suggestions for further research.

Chapter 2 - *Fuzzy Logic and Fuzzy Control*

2.1. Introduction

2.1.1. General Introduction

Since the introduction of fuzzy logic by Lofti A. Zadeh in 1965 [13], fuzzy control has developed into the most important and widespread application of this novel mathematical theory [1,2,3]. The main advantage offered by fuzzy control is the possibility of representing human linguistic expert knowledge in a machine processable format. Precisely this capability has enabled enhanced automated control of systems, where classical control strategies have failed to improve upon the performance of the human operator. Such systems are often poorly understood and thus no detailed and accurate mathematical model of the system exists. As in the case of classical control theory, the variety of fuzzy control strategies is quite broad, from the straightforward and direct rulebase approach to the more complicated methods such as adaptive combinations of neural networks and fuzzy logic. The *field of adaptive fuzzy control* can thus be classified into three groups :

- *Neural-Fuzzy Methods* - combinations of artificial neural networks and fuzzy algorithms,

- *Hybrid Adaptive Fuzzy Controllers* - combinations of fuzzy algorithms and classical control strategies and

- *Stand Alone Adaptive Fuzzy Controllers* - based only on fuzzy algorithms.

This chapter presents an outline of adaptive fuzzy control paradigms with emphasis placed on *Stand-Alone Adaptive Fuzzy Controllers*.

As in the case of the classical adaptive controller, the adaptive fuzzy controller is able to maintain and even improve its control performance through either adaptation of its structure or of its parameters. Whether this adaptation is performed *on-line* or *off-line* is a key issue.

The majority of the *off-line* methods require a mathematical model of the plant in order to support the adaptation [14,15,16]. Consequently, if the fuzzy controller developed by an off-line method is to provide reliable control of the plant, then the mathematical model of the plant, upon which the off-line adaptation depends, must be

accurate. This drawback is offset by the fact that the complexity of the adaptation can be more or less unlimited.

The *on-line* adaptive fuzzy controller strategies offer the advantage that the adaptation is carried out directly on the real plant, thus helping to ensure accurate controller performance. The main disadvantage of these on-line methods is that the adaptation algorithm must be computationally efficient. As the one of the criteria for this research is practicle, reliable and accurate controller performance, the *on-line* adaptive fuzzy control strategies are highlighted in this chapter.

As in classical adaptive control, both *direct* and *indirect forms of adaptive fuzzy control* have been developed.

In the *direct form*, the parameters of the controller are directly adapted in order to reduce the closed loop control error. This adaptation is based on the evaluation of the current controller performance using some form of performance index [17, 18]. The *indirect forms* of adaptive control strategies first estimate a model of the plant and then adjust the controller parameters based on the assumption that the estimated model is accurate [19, 20]. Eighty percent of all literature found during this research relates to the direct adaptive fuzzy control methods, which thus dominate the area of stand-alone adaptive fuzzy control. Figure 2.1, shown below, gives an overview of the adaptive fuzzy control strategies which are investigated in most detail in this thesis.

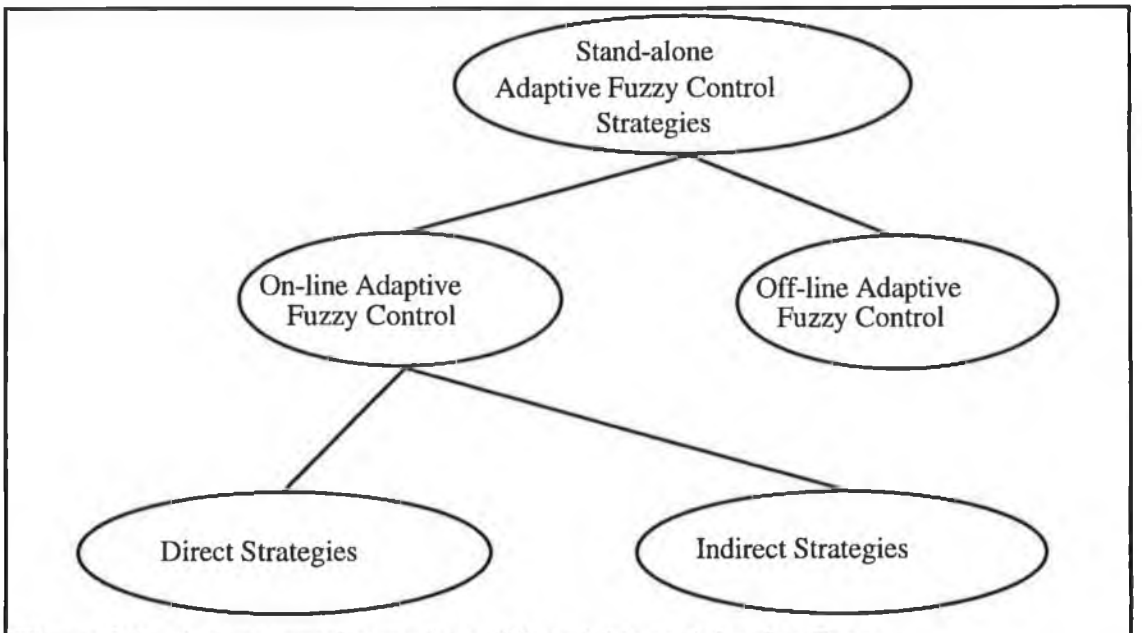


Figure 2.1 - A conceptual overview of Stand-Alone Adaptive Fuzzy.

2.1.2. Overview of Chapter Structure

Initially, *Section 2.2.* of this chapter provides the reader with a simple explanation of the principle of fuzzy logic using an everyday example. Having introduced fuzzy logic, summaries of four common computer aided software engineering (CASE) tools for fuzzy logic systems are detailed in *Section 2.3.* Due to the considerable processing requirements of larger scale fuzzy logic systems and to make real time fuzzy control feasible for highly complex and dynamic plants, specialised fuzzy hardware is available. An overview of some of the more common fuzzy hardware components is presented in *Section 2.4.* The terminology and structure of fuzzy controllers are explained and summarised by *Section 2.5.* using the example of a Proportional-Derivative Fuzzy Logic Controller. The next Sections stem from a comprehensive literature survey and are thematically divided as follows :

- *Section 2.6.* - Fuzzy Modelling,
- *Section 2.7.* - Direct Adaptive Fuzzy Control,
- *Section 2.8.* - Indirect Adaptive Fuzzy Control,
- *Section 2.9.* - Hybrid Fuzzy Control.

Each of these sections gives a synopsis of its corresponding area and relates any relevance to the warm water process. In conclusion, a summary of the chapter and concrete suggestions for possible control strategies for the warm water plant are detailed in *Section 2.10.*

2.2. Principles of Fuzzy Logic

Fuzzy logic was first developed by Lofti Zadeh in 1965 [13]. The intention of Zadeh was to provide a mathematical concept for the processing of imprecise data. Based on classical set theory, fuzzy logic extends the concept of bivalent set membership to allow the partial *membership* or *elementhood* of an object to a set. This can be illustrated by the example of classifying water temperature into hot and cold sets. Classical set theory sets a threshold value of temperature in order to differentiate between the sets of hot and cold water. If a threshold or *crisp* value of 50 °Celsius is applied then water with a temperature of 49.5 °Celsius is cold whereas water with a temperature of 50.5 °Celsius is hot - obviously not a realistic classification of these two temperatures. By applying fuzzy logic and thus utilising fuzzy sets, a more realistic and human classification is made possible. The water with a temperature of

50 °Celsius belongs in equal measure to the fuzzy set of cold water and to the fuzzy set of hot water. Figure 2.2 contains a graphical representation of this example. This soft or *fuzzy* interpolation between two sets is achieved through the introduction of the *degree of membership* which can be any value on the continuous interval [0,1], (the upperbound of the degree of membership can be any finite value but is assumed unity for normalised fuzzy sets) where total membership corresponds to a value of unity and total non-membership to a value of zero. The degree of membership of an object to a fuzzy set is calculated by a *fuzzy membership function.* A fuzzy set *A* is defined by the equation (2.1) which is the general form taken from Tilli [1].

$$A=\{x, (x,\mu_A(x)) \mid x \in X\} \tag{2.1}$$

where X is collection of objects denoted generically by x,
A is a fuzzy set in X and
$\mu_A(x)$ is the fuzzy membership function of x in A which maps X to the membership space M.

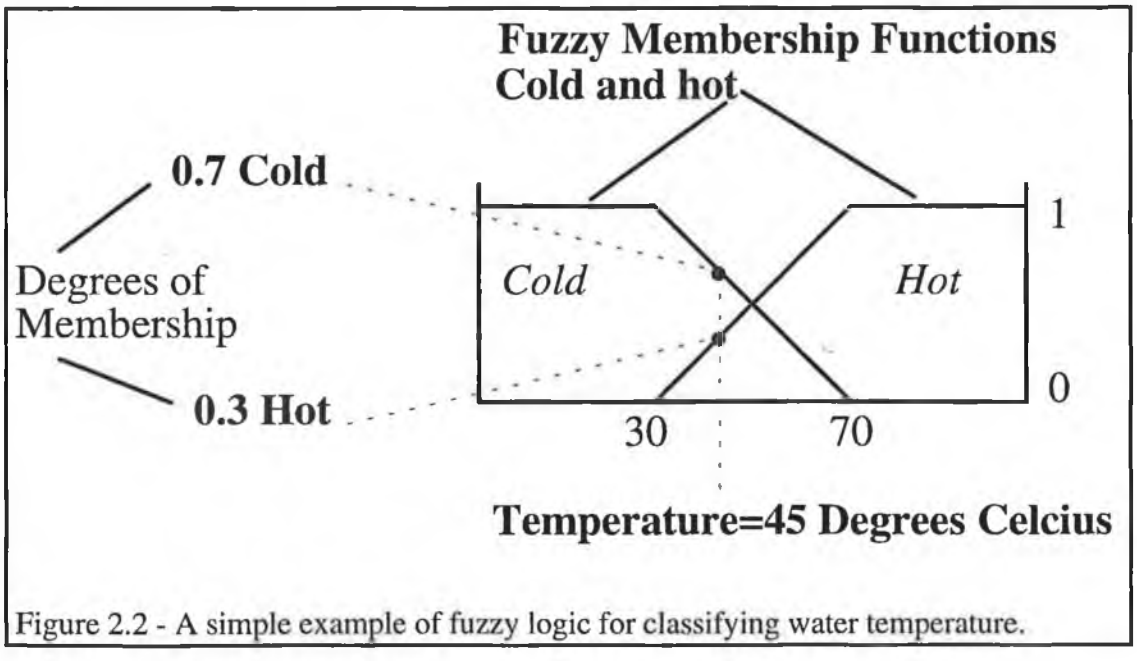


Figure 2.2 - A simple example of fuzzy logic for classifying water temperature.

The *domain* of the complete set of fuzzy membership functions is referred to as the *universe of discourse* - in the example in Figure 2.2 this is *temperature* from 0 to 100 degrees Celsius. Thus the *fuzzy membership function* converts a scalar (crisp) value into a fuzzy variable with its corresponding degree of membership. More detailed mathematical descriptions of fuzzy logic and its mathematics can be found in Tilli [1], Zimmerman [2] and Pedrycz [3].

2.3. Fuzzy Logic Software Tools

There is a *large number of CASE tools* available, for the design and simulation of fuzzy logic systems. The majority of these products are summarised in Angstenberger [21]. During the course of this research four software tools for fuzzy logic systems were used, all of which are listed below :

- *Cubicalc* - a relatively cheap and easy to use program for Microsoft Windows 3.1 that allows the design and simulation of fuzzy algorithms through a graphical user interface. The developed fuzzy system can then be compiled into BASIC, PASCAL or ANSI C source code. This program is most suitable for educational purposes [22].

- *TILShell* from Togai Infralogic - a professional fuzzy system software design tool for Microsoft Windows 3.1 that allows the development of fuzzy systems through a graphical user interface. A large range of compilers is available e.g. ANSI C, PASCAL. The package can be combined with Togai Infralogic's in-house fuzzy chip, the FC110-DCP to develop complete software/hardware solutions [23].

- *FuzzyTech* from Inform GmbH - also a professional CASE tool for the development for the fuzzy systems runs under Microsoft Windows 3.1. This software allows the ompilation of the developed fuzzy algorithm into ANSI C source code. The compiled ANSI C source code is, however, not easily adaptable as the fuzzy functions are contained within C library functions [24].

- *RT Fuzzy from MATRIXx* - is part of the MATRIXx simulation environment and thus benefits from the functionality of this leading simulation tool [25]. In contrast to the other three fuzzy CASE tools MATRIXx runs on a UNIX platform.

While all of these CASE tools have their merits, it was found that they all lack the necessary flexibility for research into adaptive fuzzy systems. Thus these fuzzy logic tools were used for general experimentation and training.

2.4. Fuzzy Hardware

The term "fuzzy hardware" refers to all hardware components that utilise specialised circuitry to accelerate the calculation of fuzzy algorithms [26,27]. There are three main types of fuzzy hardware, which can be described as follows :

- *FASIC* - **F**uzzy **A**pplication **S**pecific **I**ntegrated **C**ircuit is, as in the case of all ASICs, a relatively expensive solution and is only viable for large production volumes.

- *Stand alone Fuzzy Processors* - have the advantage of very high processing speeds for fuzzy algorithms. The disadvantage with such systems is that a completely new hardware structure is required and thus time to market is lengthened. In addition, software support is often lacking. The FP family of fuzzy processors from the Japanese company OMRON constitutes one example of stand alone fuzzy processors.

- *Fuzzy Coprocessors* - can be added to a host hardware system and thus used to relieve the host system of the fuzzy processing burden. Compared to the stand alone systems, the coprocessor approach requires little hardware reconfiguration of the complete system. The main disadvantage is the increase in required space. The FC110-DCP from Togai Infralogic can be used either as a stand alone solution or as a fuzzy coprocessor [28].

- *Fuzzy Microcontrollers* - are traditional microcontrollers with an additional on-chip fuzzy processor. This solution offers a fast and simple update of an existing microcontroller to a hardware fuzzy system. One example is the Fuzzy 80C166 from Siemens AG which utilises an optimised fuzzy inference machine from Inform GmbH. This chip is pin compatible with the 80C166 microcontroller and can replace the older non-fuzzy versions without hardware modification.

The use of fuzzy hardware is, however, only necessary when the existing non-fuzzy hardware cannot process the required fuzzy algorithms quickly enough. As the sampling time for the warm water process was thirty seconds (see Chapter 4), no fuzzy hardware was necessary for this research.

2.5. Fuzzy Logic Control

2.5.1. Areas of Application

There are three main areas of application for fuzzy logic :

- *control and automation*,
- *pattern recognition* and
- *expert systems*.

Of these, the area of *control and automation* is the most expansive [1,2]. This section gives an introduction to the main issues of fuzzy control and defines the terminology used within this research. In order to achieve this, later on in this Chapter the example of the Proportional-Derivative Fuzzy Logic Controller is utilised.

2.5.2. Introduction to Fuzzy Logic Control

The Fuzzy Logic Controller (FLC) allows the control engineer to develop a controller for a particular process based on linguistic expert knowledge. The capability of fuzzy logic controllers to represent a heuristic control strategy contrasts with that of a classical deterministic controller where algorithms based on exact and deterministic mathematical models of the plant to be controlled are utilised. Many successful implementations of FLCs have been reported recently for non-linear processes such as wastewater treatment [29], heat exchangers [30], underground train systems [31], home heating systems [32] and applications in the automobile industry [33]. Moreover, the area of consumer electronics has benefited dramatically through the application of fuzzy logic to control tasks in such products as washing machines, rice cookers, video cameras and television tuners [1].

The following highlights the main advantages and main problems of fuzzy logic control, which are categorised below [34,35] :

Advantages

- *No mathematical plant model required* - the rulebase approach of the fuzzy logic controller allows the control of a plant based on expert knowledge. Thus a plant operator can be interviewed and the acquired knowledge stored in a rulebase which forms an integral part of the fuzzy controller.

- *Possible linguistic description* - the expert knowledge can be described linguistically through the use of fuzzy sets with labels such as *small*, *tall*, *positive large* etc. for each controller input and output variable. This allows a human interpretation of the controllers actions.

- *Non-linear control* - the fuzzy logic controller is a non-linear controller which performs interpolation between the rules contained within the rulebase. The form of this interpolation is determined by the membership functions, the inference functions used in the rules

and the defuzzification method. The fuzzy logic controller thus has the capability of controlling non-linear plants around various operating points.

- *Robustness* - some investigations [3,36] have claimed that the fuzzy logic controller is more robust to process noise and parameter variance than a classical controller designed for a single operating point.

- *Little in-depth knowledge of control theory needed* - due to the rulebase structure of the FLC, a controller can be designed by an engineer without in depth knowledge of classical closed loop control theory.

Problems

- *Lack of formality* - there is no formal method for the design of the FLC for a particular control problem and thus the engineer must resort to a combination of experience and trial and error, which can lead to lengthy commissioning and non-optimal solutions.

- *Lack of stability proof* - there is no general method for proving the stability of the FLC. This disadvantage has led to a mixed reception for the FLC within the closed loop control theory community [37].

- *Large number of degrees of freedom* - where as the PID controller, for example, has three degrees of freedom, the fuzzy controller has many more, all of which effect the response of the controlled system. This can lead to a lengthy and complicated subjective optimisation of the fuzzy controller.

- *Acceptance problems*- the heuristic nature of the fuzzy controller, the marketing strategy of "fuzzy products" combined with the exaggerated claims by some of its proponents have lead to acceptance problems within the control theory community [37].

2.5.3. Structure and Terminology of the FLC

In order to gain familiarity with the structures and terminology of the FLC the *example of a Proportional-Derivative FLC (PD-FLC)* is explained in this section. Although the classical linear PI controller is mostly used in industry, the PD-FLC is most often implemented fuzzy controller. The structure of the PD-FLC is shown in Figure 2.3 and is described in more detail by the following points :

- *Input Variables* - the error and first derivative of the error are utilised for the PD-FLC. This approach is similar to the phase plane form of representation of a control system. The engineer converts the phase plane to a partitioned fuzzy space and enters the corresponding actions in the corresponding rulebase cells.

- *Input Variable Gains* - (*Ke*, *Kde*) these gains are used to map the input variables to their respective universes of discourse within the fuzzy controller. The dynamics of the controller are dependent on the values of these gains [3].

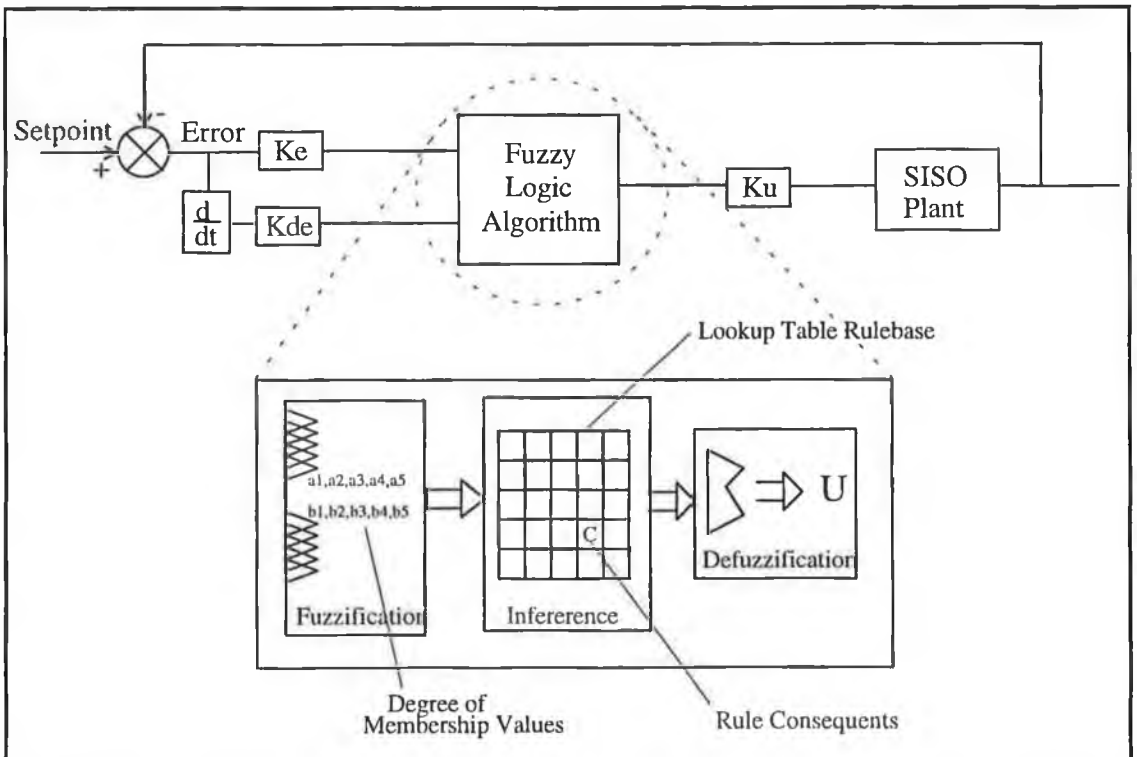


Figure 2.3 - Block diagram of a Proportional Derivative Fuzzy Logic Controller (PD-FLC)

- *Fuzzification* - converts the crisp input values to fuzzy variables. This is achieved by calculating the degree of membership of each fuzzy membership

set for a given value of the input variable. This process is shown in Figure 2.2 (see page 12) for the water temperature example. In Figure 2.3 (see page 17) the degrees of membership for the input variable *error* are depicted by a_1, a_2, a_3, a_4 and a_5.

- *Inference functions* - the functions used to evaluate the linguistic logical functions such as AND and OR. These functions belong to the so-called T-Norms and Co-T-Norms (or S-Norms). Details of some of these functions and their properties can be found in Zimmerman [2]. The functions used in this research are :

    1. *Minimum* - $\mu_{a,b} = \min(\mu_a, \mu_b)$ corresponding to the linguistic AND function,

    2. *Product* - $\mu_{a,b} = \mu_a \times \mu_b$ corresponding to AND,

    3. *Maximum* - $\mu_{a,b} = \max(\mu_a, \mu_b)$ corresponding to OR,

    4. *Algebraic Sum* - $\mu_{a,b} = \mu_a + \mu_b - \mu_a \times \mu_b$ corresponding to OR,

    5. *Fuzzy-OR* - $\mu_{a,b} = \gamma \max(\mu_a, \mu_b) + \frac{1}{2}(1-\gamma)(\mu_a + \mu_b)$ an adaptable form of the linguistic OR function where γ is a weighting parameter that allows the user to adjust the function between the extremes of the maximum (γ=1) and mean functions (γ=0),

    6. *Fuzzy-AND* - $\mu_{a,b} = \gamma \min(\mu_a, \mu_b) + \frac{1}{2}(1-\gamma)(\mu_a + \mu_b)$ an adaptable form of the linguistic AND function, where γ is a weighting parameter that allows the user to adjust the function between the extremes of the minimum (γ=1) and mean functions (γ=0), and

    7. *Mean* - $\mu_{a,b} = \frac{\mu_a + \mu_b}{2}$.

    where μ_a and μ_b are degrees of membership and $\mu_{a,b}$ is the inferred degree of membership.

- *Rulebase* - There are two main types of rulebases - the *lookup table* and the *fuzzy relational matrix.*

In Figure 2.3 (see page 17) a lookup table rulebase has been utilised within the PD-FLC. Each element in a lookup table contains the *consequent* (defined below) part of the rule. Additionally, each element is indexed by the *antecedent* (defined below) variable sets that are active i.e. where the degree of membership is greater than zero. The fuzzy relational matrix is a memory intensive matrix that contains possibility values for all possible rules for a given fuzzy algorithm [2]. Where the lookup table rulebase gives a single fuzzy membership set for a particular antecedent configuration, the fuzzy relational matrix delivers a complete possibility distribution over the consequent universe of discourse. Due to the large memory requirements, the fuzzy relational matrix is not often used.

The mathematical representation of a lookup table rulebase is shown below:

$$R_1 \rightarrow \textbf{\textit{If}}\ X_1\ is\ A_{1,X_1} \otimes X_2\ is\ A_{1,X_2} \otimes \ldots\ldots \otimes X_n\ is\ A_{1,X_n}$$
$$\textbf{\textit{Then}}\ Y_1\ is\ B_{1,Y_1}\ and\ Y_2\ is\ B_{1,Y_2}\ and \ldots\ldots and\ Y_m\ is\ B_{1,Y_m}$$
$$R_2 \rightarrow \textbf{\textit{If}}\ X_1\ is\ A_{2,X_1} \otimes X_2\ is\ A_{2,X_2} \otimes \ldots\ldots \otimes X_n\ is\ A_{2,X_n}$$
$$\textbf{\textit{Then}}\ Y_1\ is\ B_{2,Y_1}\ and\ Y_2\ is\ B_{2,Y_2}\ and \ldots\ldots and\ Y_m\ is\ B_{2,Y_m}$$
$$\bullet$$
$$\bullet$$
$$\bullet$$
$$R_k \rightarrow \textbf{\textit{If}}\ X_1\ is\ A_{k,X_1} \otimes X_2\ is\ A_{k,X_2} \otimes \ldots\ldots \otimes X_n\ is\ A_{k,X_n}$$
$$\textbf{\textit{Then}}\ Y_1\ is\ B_{k,Y_1}\ and\ Y_2\ is\ B_{k,Y_2}\ and \ldots\ldots and\ Y_m\ is\ B_{k,Y_m}$$

where n is the number of input variables,
m is the number of output variables,
k is the number of rules R,
A and B are the input and output variable fuzzy membership sets respectively and
$\otimes$ is an inference function.

Each rule consists of a *premise* or *antecedent* part :

$$\textbf{\textit{If}}\ X_1\ is\ A_{1,X_1} \otimes X_2\ is\ A_{1,X_2} \otimes \ldots\ldots \otimes X_n\ is\ A_{1,X_n}$$

followed by an *action* or *consequent* part :

$$\textbf{\textit{Then}}\ Y_1\ is\ B_{1,Y_1}\ and\ Y_2\ is\ B_{1,Y_2}\ and \ldots\ldots and\ Y_m\ is\ B_{1,Y_m}$$

The consequent variables (Y_1 ... Y_m) of the PD-FLC can be either fuzzy membership sets or fuzzy singletons. A fuzzy singleton is a scalar (crisp) value.

The mathematical representation for the relational matrix form of a rulebase is described below for two antecedent single consequent fuzzy system :

$$R_1 \rightarrow \textit{If } X_1 \textit{ is } A_{1,X_1} \otimes X_2 \textit{ is } A_{1,X_2} \quad \textit{Then } Y_1 \textit{ is } B_{1,Y_1} \Rightarrow g_{1,1,1}$$

$$R_2 \rightarrow \textit{If } X_1 \textit{ is } A_{1,X_1} \otimes X_2 \textit{ is } A_{1,X_2} \quad \textit{Then } Y_1 \textit{ is } B_{2,Y_1} \Rightarrow g_{1,1,2}$$

$$\vdots$$

$$R_l \rightarrow \textit{If } X_1 \textit{ is } A_{1,X_1} \otimes X_2 \textit{ is } A_{1,X_2} \quad \textit{Then } Y_1 \textit{ is } B_{l,Y_1} \Rightarrow g_{1,1,l}$$

$$\vdots$$

$$R_{pq+1} \rightarrow \textit{If } X_1 \textit{ is } A_{p,X_1} \otimes X_2 \textit{ is } A_{q,X_2} \quad \textit{Then } Y_1 \textit{ is } B_{1,Y_1} \Rightarrow g_{p,q,1}$$

$$R_{pq+2} \rightarrow \textit{If } X_1 \textit{ is } A_{p,X_1} \otimes X_2 \textit{ is } A_{q,X_2} \quad \textit{Then } Y_1 \textit{ is } B_{2,Y_1} \Rightarrow g_{p,q,2}$$

$$\vdots$$

$$R_{pq+l} \rightarrow \textit{If } X_1 \textit{ is } A_{p,X_1} \otimes X_2 \textit{ is } A_{q,X_2} \quad \textit{Then } Y_1 \textit{ is } B_{l,Y_1} \Rightarrow g_{p,q,l}$$

where p and q are the number of antecedent fuzzy membership sets for X_1 and X_2 respectively,
l is the number of consequent fuzzy membership sets for Y_1 and $g_{i,j,k}$ (i=1 to p, j=1 to q, k=1 to l) is the possibility value for each of the complete set of possible rules.

- *Rule activation value* - the inference functions contained within the antecedent part of a rule act on the degrees of membership of the antecedent

fuzzy membership sets. The calculated value is the *rule activation value* and thus reflects the strength of the rule.

- *Aggregation* - if two or more active rules infer the same consequent fuzzy membership set, then an aggregation function is used to combine the activation levels of these rules in order to calculate the final degree of membership for the inferred consequent fuzzy membership set. The most common are *mean* and *maximum* [38].

- *Defuzzification* - is the method by which the fuzzy output from the complete rulebase inference is converted to a crisp value. There are over thirty methods of defuzzification to be chosen from. The most common are *Mean of Area*, *Maximum Height* and the *Centre of Gravity* [38]. An adaptive defuzzification method that allows an adjustment between the centre of gravity method and the maximum of height is developed in Kiendl [39].

- *Output Variable Gain Ku* - this is applied to the crisp output value from the fuzzy controller. As in the case of the input gains, the system dynamics are effected by its value [3].

This section has illustrated some of the key structural aspects of an FLC and defined the terminology used in this research. Other forms of the FLC for SISO systems are possible. One example is the Proportional-Integral form Fuzzy Logic Controller (PI-FLC), which is formed by placing an integrator between the output of the PD-FLC and the plant. The reader is referred to references [38] for more detailed and illustrative descriptions of fuzzy control.

It should be noted that the issue of standardising the terminology within the field of fuzzy control has not been addressed. Currently, the "Verein Deutscher Ingenieure VDI " is working together with the IEEE to standardise the terminology within both the English and German languages [40].

2.6. Fuzzy Modelling

Just as a control strategy for a particular plant can be linguistically described and programmed as a set of rules within a fuzzy algorithm, so to can the dynamic behaviour of the plant. This process of linguistic modelling is termed fuzzy modelling. This section gives a brief overview of the field of fuzzy modelling.

The most important characteristic of fuzzy models is that they are *universal approximators*, thus allowing them to model any function to an arbitrary degree of accuracy [11]. There are two main types of fuzzy models which differ in the type of consequents used in the rulebase [41]. These are as follows :

- *Function Consequents* - this type of fuzzy model utilises a function of the input variables for each consequent. Thus an interpolation between the functional consequents of active rules is performed by the fuzzy algorithm. One example could be the modelling of a non-linear system with first order dynamic behaviour. The fuzzy algorithm could utilise first order ARX models of the system interpolated across different operating points as consequents.

- *Fuzzy Variables or Singleton Consequents* - this is the simpler of the two options, where the consequent variables of the fuzzy model algorithm are either fuzzy membership sets or fuzzy singletons. These models can thus utilise either the look-up table or relational matrix formats for the rulebase.

Pioneering work in the field of fuzzy modelling has been performed by Pedrycz [3] and Tong [42] whose models utilise fuzzy variable consequents in relational matrix form. Takagi and Sugeno [43] developed functional consequent fuzzy models and applied them to the modelling and control of a multilayer incinerator [44]. More recently Küpper [45] has used stochastic approximation to determine the possibility values of rules in a fuzzy model with a relational matrix rulebase format.

2.7. Direct Adaptive Fuzzy Control

2.7.1. Introduction

A direct adaptive fuzzy controller utilises state data from the plant in conjunction with some form of performance index to adapt the controller parameters directly. For the on-line methods, no model of the plant to be controlled is included in the controller structure.

As previously described the fuzzy controller is a non-linear controller and thus suited for the control of non-linear systems. The introduction of an adaptive mechanism helps the controller to maintain and even improve control performance for non-linear or especially time varying systems.

2.7.2. Types of Adaptation

The first type of adaptive mechanism considered in this section is *Gain Tuning/Adaptation* (GTA) [46]. This form of adaptation is similar in type to that of the self-tuning PID controller. GTA methods assume that the controller possesses a sufficient and correct rulebase for control of the plant around at least one operating point. This assumption is disadvantageous, as one destructive rule can hinder good controller performance no matter what gain values are chosen. The adaptation performed is of a linear type, as the non-linear characteristic of the rulebase which generates non-linear controller actions is not altered. Because no complete rulebase could easily be developed, application of this strategy for the control of the warm-water process with its multi-variable non-linear coupled nature would be limited.

The second type of adaptation is that of *rule modification,* initially termed *Self Organising Control* (SOC) by Procyk and Mamdani [17]. Both the consequent and antecedent terms within a rule can be manipulated. This form of adaptation is more flexible than that of GTA and can be considered to be non-linear. One important aspect of this adaptation strategy is that destructive rules can be corrected and replaced. Moreover, it is possible for the rulebase to be partially or even completely empty in the initial stages of controller operation. This form of controller is certainly more suitable for the control of the warm water process than the GTA methods, due to the flexible adaptation and the lack of the requirement for a complete initial rulebase.

The third type of adaptation is that of *Membership Set Modification.* Both the membership sets of the input and output variables can be manipulated. Due to the large number of parameters that are involved the adaptation process is typically time consuming and some form of optimisation strategy suited to large dimensional problems such as simulated annealing or evolutionary algorithms is required [47]. Moreover, this controller type is usually adapted off-line. This method also has the disadvantage that a good rulebase is assumed to be available.

Most of the literature encountered during this research describes control strategies for *Single Input Single Output (SISO) systems.* Moreover, few of the publications validated their strategies with real world control problems. Most demonstrations were limited to the control of well known plants within a simulation environment. Often no comparison with a classical controller was made and results were not fully explained with issues such as stability, processing and memory requirements being conveniently ignored. It thus remains to be seen if any of the direct adaptive fuzzy controller archetypes that have been described for SISO systems can be modified and

extended for successful control of the *Multi Input Multi Output (MIMO) warm water process.*

2.7.3. Gain Tuning/Adaptation

Gain Tuning/Adaptation (GTA) adapts or tunes some or all of the gains for the input and output variables of a fuzzy controller. In Figure 2.3, (see page 17) containing the PD-FLC these gains are *Ke*, *Kde* and *Ku*. The aim of this adjustment is to match the dynamic and steady state response of the FLC based system to that of a desired response.

One of the simplest mechanisms for gain tuning is that of Bare et al [46]. They develop a FLC with gain adaptation to control a gasoline catalytic reformer. The gains KE and KDE were adapted using a so-called simple crisp heuristic. The assumption was made that the larger *Ke*, the faster but more oscillatory the response. The gain *Ku* is not adjusted as this could cause instability. The crisp heuristic used in this publication is given by (2.2, 2.3, 2.4).

$$\rho = \frac{e(k)}{e(k-1)} \tag{2.2}$$

1. If $|\rho| > \alpha$ then the gain *Ke* is incremented (undershoot) or decremented (overshoot), where α is a user defined constant. (2.3)

2. If $\rho^2 > \beta$ then the gain K*de* is incremented or decremented, where β is a user defined constant. (2.4)

The controller thus created was deemed to be capable of maintaining good control over a wide range of operating points. No comparisons with a standard controllers were detailed.

Daugherity et al [36] develop a simple auto tuner for the *Ke* and *Kde* gains of a PD-FLC for the control of an industrial gas fired water heater. Comparisons were made with a PID controller. The auto tuning PD-FLC was shown to be able to control the plant as well as a PID controller for the tuned operating point and to exhibit superior robustness in the presence of system noise and for various operating points.

In a more complex method, Zhao et al [48] have designed a fuzzy tuner to modify the gains *Ke*, *Kde* and *Ku* of a PD-FLC. The tuning mechanism utilised is a simple fuzzy

algorithm. The *fuzzy tuner* has performance indices such as overshoot, settling time and rise time as input variables and incremental changes to the gains as output variables. Each of the rules can described as follows :

$$\text{If } OS \text{ is } A_i \text{ and } RS \text{ is } B_i \text{ and } ST \text{ is } C_i \text{ then } \Delta Ke \text{ is } D_i \text{ and } \Delta Kde \text{ is } E_i \text{ and } \Delta Ku \text{ is } F_i \tag{2.5}$$

where OS is overshoot %,
RS is the rise time in seconds divided by the dominant plant time constant,
ST is the settling time divided by the rise time and
A_i, B_i, C_i, D_i, E_i, F_i are fuzzy membership sets.

An adaptation rate parameter α is also introduced for all gains in the following form :

$$\text{Gain}(n+1) = \text{Gain}(n) - \alpha\Delta\text{Gain}(n) \tag{2.6}$$

where n is an iteration variable.

Claims are made that the fuzzy tuner brings some performance rewards for off-line tuning and on-line adaptation to systems with time variance. The issue of stability is not considered.

To successfully apply the concept of GTA to the warm water process the following problems would have to overcome :

- *availabilty of complete and correct rule bases* for each of the controlled variables,
- *decoupling of the variables* would have to be at least partially realised,
- using the two dimensional FLC approach, there would be *four to six gains to be adjusted* and
- *introduction of stability criteria* necessary to monitor the adaptation of the input and output variable gains.

Although simulation may yield reasonable results for SISO systems and fully decoupled multi-variable systems, due to the above listed requirements, the GTA adaptive fuzzy controller is deemed not to be suited for the control of the warm water process.

2.7.4. Adaptation of Rule Consequents

Details of the first rule adaptive fuzzy controller were published in 1979 by Procyk and Mamdani [17]. This controller was termed a *Self-Organising Controller* (SOC). It is claimed that the SOC analyses its control performance and through adaptation of the rule consequents is able to achieve some improvement in its control performance. This form of adaptation is non-linear and is thus more flexible than the previously described GTA methods.

The SOC presented by Procyk and Mamdani is based around a PD-FLC, see Figure 2.4, and contains three extra elements :

- a *performance index,* whereby although used in all the SOC literature found, the term "performance index" is not strictly correct but rather the term "reference model" is more suitable,
- an *incremental plant model* and
- a *rule modification algorithm.*

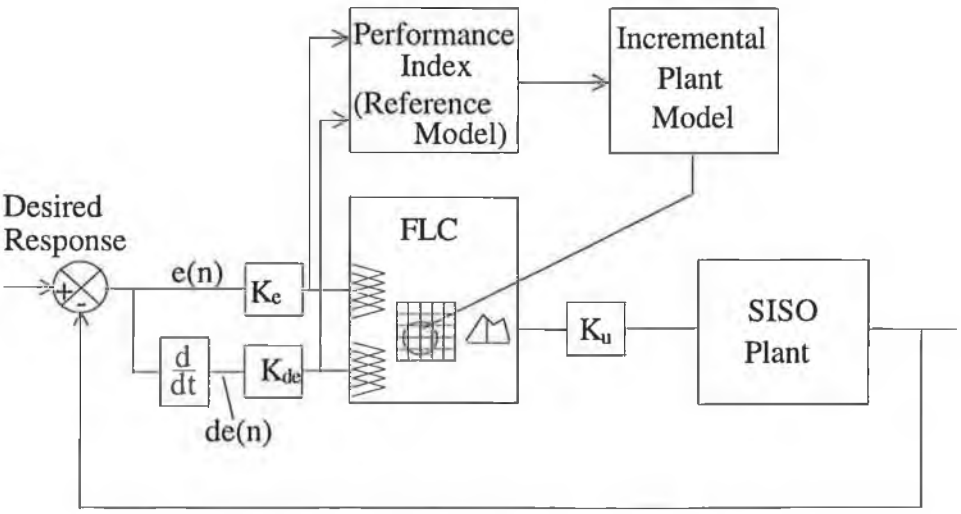


Figure 2.4 - Block diagram of a Self-Organising Controller (SOC).

The *performance index* (or more correctly reference model) is used to analyse the controller performance and serves as a reference for the desired response in order to adapt the rulebase. Its inputs are scaled error *Ke.e(n)* and rate of change of error *Kde.de(n)* which serve as a measure of the deviation of the controlled variable from the desired trajectory. The performance index output *p(n)*, gives an indication of controller performance and is used together with an incremental process model to implement rule adaptation. The performance index can be expressed in the form of a simple look-up table or as a set of rules in a fuzzy rulebase. According to Procyk and Mamdani, the performance index is not process specific as it represents the "minimum tolerable closed loop response of the system".

The *incremental plant model* used by Procyk and Mamdani is the plant Jacobi matrix. In practice, this incremental plant model maps the output of the performance index to the rulebase adaption block, thus acting as a learning gain. Chapter 5 contains some simulations of the SOC algorithm within which the value of the incremental plant model is seen to effect the learning rate of the rulebase.

From this point on in this research the term "reference model" will be utilised instead of the original "performance index". The reason being that the SOC "performance index" does in fact function as a reference model for the desired controller behaviour, thus resembling the MRAC controller.

The adaptation of the *rule modification algorithm* is a credit assignment problem, which involves reinforcement or correction of the controller rules that contributed to the present controller performance. This present controller performance is, however, the result of some controller action that occurred in the past. The exact point in time in which this action was carried out is related to the dynamics of the system and is said to have occurred *m* samples in the past. Thus adaptation is carried out on the consequents of the active rules *m* samples in the past. The output of the reference model is fed to the normalised incremental process model of the plant, whose output is then used to modify the rule consequents. Based on experimentation, claims are made that the SOC is insensitive to simple or even incorrect incremental process models. The algorithm used for rule modification is given by equation (2.7) which is quoted directly from the publication :

$$R(n+1)=\{R(n) \textbf{ and not } R'(n)\} \textbf{ or } R''(n) \qquad (2.7)$$

where ***and not*** and ***or*** are linguistic operators,

R(n) is the current rulebase,
R(n)' is the old rulebase responsible for current control performance,
R(n)'' is the desired rulebase and
R(n+1) is the new adapted rulebase.

This method of rulebase adaptation creates three new rules for every rule change and thus some form of rule deletion mechanism is needed if this method is to be used.

Experiments in simulation were carried out using *empty* initial rule bases for SISO and MIMO systems with and without process dead time and in the presence of noise. The incremental models and input and output variable gains values were all selected to give a good response - no further details are given as to how these values were chosen. Although convergence was, in some cases never achieved, all responses were of a reasonable nature, with a steady state error of less than 10% of the setpoint value. The controller response typically had a steady state offset for which no reason is given. It is more than likely due to the interpolative nature of fuzzy controllers, which interpolate between the rules in the rulebase.

Sugiyama [49] further analysed some aspects of the SOC whereby the following improvements and additions to that of Procyk and Mamdani [17] were achieved :

- *PID type of FLC* is used to increase control performance. The inputs are error, first and second derivatives of the error,

- *non-linear mapping* of all input variable fuzzy membership sets to improve rise time and sensitivity around the setpoint,

- some *guidelines* for the development of the reference model are presented and an analogy is made between the reference model and the model of the desired response in the classical model reference adaptive controller (MRAC),

- a *design method* for a SOC is briefly described with a suggested relationship between the FLC gains *Ke* and *Kde* and a first order system and

- the *accuracy of the delay parameter m*, i.e. which rulebase contributed to the current control performance, is deemed to be unimportant.

Linkens and Abbod [18] clearly describe the development of a SOC based on the relational matrix structure of Procyk and Mamdani [17]. A gain switch is used to increase the controller sensitivity around the set point and details are given for the calculation of the gains *Ke*, *Kde* and *Ku*. The SOC algorithm is applied to a coupled tanks apparatus and to coupled motors.

Procyk and Mamdani [17], Sugiyama [49], Shao [50] and Linkens and Abbod [18] all utilise a fuzzy relational matrix approach to rule modification which, for MIMO systems, is computationally impractical. This large computational burden is quantified in Wakileh and Gill [51], where the application of the SOC to robot motion control is reported. In order to reduce the necessary processing time fuzzy singletons were adopted.

Ho and Lin [52] suggest the use of a simple lookup table to replace the relational matrix. The rule adaptation algorithm for a SISO system can be mathematically described as follows :

$$R_k(n+1) = R_k(n-m) + \alpha\, p(n) \tag{2.8}$$

where $p(n)$ is the reference model output value,
α is the learning gain,
m is the delay in learning and
R_k refers to a single rule.

This adaptation is thus a weighted update, whereby the parameter α directly effects the learning rate. In addition, Ho and Lin suggest the utilisation of conditioned learning mechanism, which prevents the SOC from adaptation if its trajectory is already satisfactory. A good description of the reference model table is given as well as a possible modification which increases the rate of learning of the SOC.

Farbrother, Stacey and Sutton [53] apply a SOC to the control of a remotely operated submersible. Their application is the standard Procyk and Mamdani SOC with the exception that a lookup table rulebase instead of the relational matrix form is used and two meta rules are added to assist advantageous learning :

1. $Rule\{\, e(n)=0\,,\ de(n) = 0\,\} = 0$ - ensures equilibrium at the setpoint. (2.9)

2. $Rule\{\ e(n)=x\ ,\ \ de(n)\ =\ y\ \}\ =\ -Rule\{\ e(n)=-x\ ,\ \ de(n)\ =\ -y\ \}$ - ensures a symmetrical rulebase thus helping to anticipate overshoot. (2.10)

Spinrad [54] suggests that the use of a reference model table gives unpredictable control as no specific method for the development of the reference model for a given response has been developed. Instead he utilises the difference between the actual error and a desired error as the basis for rule consequent manipulation. The desired error can, for example, be some function of the present error. In addition, the use of a *rule importance factor is* introduced. This replaces the time delay factor *m* by weighting the past rules over several samples in the past. Although Spinrad's approach to the SOC is simpler than that of Procyk and Mamdani, Sugiyama and Stacey and Sutton, it is claimed that it achieves performance equivalent to and in some instances superior to that of a well tuned PI controller for a SISO coupled tanks process.

Spinrad [55] applies the SOC to the control of a warm water process where the tank head and temperature are to be controlled under the assumption of perfect mixing. For comparison purposes a linear quadratic Gaussian controller was designed for the process. The SOC is claimed to be easier to develop than the LQG controller and offers performance equal to the LQG controller.

Burkhardt and Bonissone [14] compare a state space pole placement controller with a FLC where the rulebase consequents have been determined off-line by a SOC. The SOC used is similar to that from Procyk and Mamdani [17] but only one rule per sampling instant is adapted. To further increase the performance of the SOC generated rulebase, a gradient descent method with a variable step size is used to off-line optimise both a non-linear mapping of the membership sets and the three input and output variable gains *Ke*, *Kde* and *Ku*. The optimisation cost function used is a weighted sum of rise time, overshoot, settling time and steady state error. The plant used was an inverted pendulum for which a detailed simulation model existed. Both the optimised SOC rulebase and the non-optimised SOC rulebase are claimed to out perform the pole placement controller. It should be noted that the SOC was developed within the simulation environment. The optimisation process using the gradient descent method is not suitable for on-line control at a high sampling rate due to time constraints.

In all hitherto described implementations of the SOC, if the set point is changed, the position of the controlled variable error within the phase plane is shifted away from

the equilibrium point and rule adaptation occurs. This adaptation is irrespective of whether the error state has been moving towards the equilibrium point or not. In an attempt to improve the adaptive performance of the SOC, Zhang and Edmunds [56] developed a new adaptation method. The approach bases rule adaptation on the trajectory of the controlled variable error within the phase plane and not on its position. Thus rule adaptation occurs only when the current rulebase is unable to drive the controlled variable to the setpoint along a desired trajectory. Zhang and Edmunds claim smoother responses from this form of SOC as compared to that from Procyk and Mamdani [17].

Song and Park [57] claim to have developed an improved SOC through a slight modification of the adaptation equation that was used by Procyk and Mamdani [17]. The new algorithm is shown to offer good control for a second order system with dead time, an open loop unstable plant and a non-linear plant.

2.7.5. Adaptation of Fuzzy Membership Set Parameters

Nomura, Hayashi and Wakami [16,58] have used the gradient descent method to tune the centre values and widths of triangular antecedent sets and the values of consequent fuzzy singletons in order to optimise the off-line control performance of a FLC.

Batur and Kasparian [59] have designed a strategy through which three consequent membership functions are adapted. The adaptation strategy is based mainly on a future predicted error obtained from a process model. However, should the process model be inaccurate, the largest past error within a time window is used instead. A correlation function is used on-line to determine the process model accuracy. The three consequent membership sets are of the linear type. The slope of each is increased with small predicted or past errors and decreased with large predicted or past errors. This method requires the use of a well tuned rulebase for the process to be controlled and has only been realised in simulation.

Isaka, Sebald and Karimi [47] have applied simulated annealing to the off-line numerical optimisation of a FLC to be used for blood pressure control during surgery. This method of optimisation requires a good plant model so that the optimised controllers can exhibit good on-line control of the intended plant.

To achieve good control, Chen, Lin and Hsu [60] modify the centre points of the consequent membership functions. The learning method used is based on temporal difference which uses artificial neural network elements. The method was applied in simulation to SISO time invariant systems with some success.

Wang [61,11] utilises orthogonal gradient descent methods and the backpropagation algorithm to optimise the antecedent and consequent fuzzy membership set parameters for fuzzy models. These methods are also applicable to the off-line optimisation of a FLC.

All of the above adaptive methods have the following disadvantages :

- *only* applied in simulation to *SISO systems*,
- the *complexity of the optimisation problems* encountered were too large for on-line processing and thus off-line optimisation is necessary,
- in order to achieve reasonable on-line controller performance, an *accurate plant model* for the off-line optimisation is necessary and
- for *MIMO systems* the complexity of the optimisation problem increases significantly.

2.7.6. Combined GTA and Rule Adaptation

Gain and rule adaptation is a combination of the GTA methods and those of the Self-Organising Controller as previously described in Sections 2.7.2 and 2.7.3. The main concern with this method of adaptation is that *rules and gains are strongly interactive.* Thus erratic outputs from the resulting controller are possible. Not much research has been conducted in this area and thus the literature available is limited.

The dual adaptation mechanism developed by Mallampati and Shenoi [62] uses a simulated annealing algorithm for gain tuning. The rule adaptation is accomplished by changing look-up table rulebase consequent entries. By analysing how often a rule fires it can be determined whether a rule is effective or not. It is claimed that if a rule often fires then it must be adapted. All rules that fire above a certain "frequency" are thus adapted. Claims are made that even with positive feedback the rulebase and

gains were adapted within five setpoint change learning trials to give good position control of an electric motor. All tests were within a simulation environment.

Stipanicev, De Meyer and Gorez [63] have developed four simple heuristic rules for adaptation of the gains *Ke*, *Kde* and *Kie* of a PID-FLC. They have combined this heuristic rulebase with a SOC for the control of a two joint robot in a simulation environment. Some success was achieved with this simple method, however some difficulty was experienced when the initial SOC rulebase was empty.

A further publication by Maeda, Sato and Murakami [64] also offers other simple suggestions for the combination of gain tuning and rulebase adaptation. Both of these methods assume an initial rulebase. A simple rulebase is used to adapt the gains with previous values of overshoot, rise time and settling time serving as performance criteria. Based on simulation results for a linear second order system, some improvements of SOC performance are claimed.

The approaches to direct adaptive fuzzy control described in this section are all off-line methods and deal only with SISO systems. For the MIMO warm water process some interaction of gain adaptations for each controlled variable would be experienced. The practical value of these methods is questionable.

2.8. Indirect Adaptive Fuzzy Control

This section describes indirect daptive fuzzy controllers. These controllers utilise a model of the plant to be controlled in order to determine the values for the manipulated variable [61,11]. Very little literature was found dealing with this controller paradigm.

Graham and Newell [65] describe the development of a single step predictive controller for a SISO system. The models used in this controller predict the future change in error of the controlled plant based on the current change in error and a possible change in controller output one step into the future. A comparison of the method using two fuzzy models is given :

- a *general fuzzy model of a linear first order system* and
- an *adaptive fuzzy model of the plant* to be controlled.

The introduction given to adaptive fuzzy modelling by Graham and Newell describes and compares the fuzzy relational matrix rulebase approach and the simpler lookup table rulebase method. An important aspect of this comparison is that the lookup table rulebase method is shown to be nearly as accurate as the relational matrix while saving considerable computer resources. The applied lookup table rulebase approach is made adaptive using the learning algorithm used given by equation (2.11) to adapt the consequents of the rulebase.

$$R(k+1)=(1-\alpha)\,R(k)+\alpha\,\mu\,X(k) \tag{2.11}$$

where R is the rule consequent,
X is the value of the variable to be modelled,
α is the weighting parameter and
μ is the rule activation level.

The rate of adaptation α is chosen so as to give a good compromise between speed of learning and model robustness to process noise. The predictive controller uses the model to calculate the next error from a series of possible controller outputs and in combination with a set of meta rules chooses the controller output that gives the best error performance. One example of such a meta rule is that only controller outputs that generate error of the same sign as the current error are permissible, thus reducing overshoot. To apply this type of predictive controller to the warm water process the problem of mulitiple control goals due to interacting control variables would have to be resolved.

Moore and Harris [19] and Moore, Harris and Brown [20] apply indirect adaptive fuzzy control to the problem of ship heading regulation under actuator signal constraints. This control problem is described in the IFAC "Benchmark Problems for Control System Design", 1990. The controller concept adopted uses a fuzzy model with a fuzzy relational matrix rulebase of the plant in question combined with a model for closed loop performance specification. Moore and Harris [19] claim to have succeeded in separating the adaptation and controller specification so that each can be separately examined and optimised. This contrasts with the previously described Self Organising Controller, described in Section 2.7.3.1, where adaptation and controller performance are intrinsically linked. The fuzzy models used by Moore and Harris are for SISO systems and thus the memory requirements of the fuzzy relational matrix are not demanding. When associative inference functions are used for the fuzzy relational matrix, the fuzzy model can be inverted. Through this inversion an ideal controller

can be theoretically created for the plant. They also examine a single step predictive fuzzy controller, similar to that of Graham and Newell [65].

If fuzzy models of the warm water process could be constructed then single step predictive fuzzy control could possibly be used to control both the outlet flow and temperature of the warm water process. As the predictive control described in this section is single step, some problems with multi-variable control of the warm water process due to the non-linear mixing characteristics could be experienced. The dynamic behaviour of the warm water process is described in more detail in Chapter 4 of this thesis.

2.9. Hybrid Fuzzy Control

The term *hybrid fuzzy controller* refers to all controller algorithms which augment some classical control strategy with a fuzzy logic algorithm. Three common examples of this archetype of fuzzy controller are :

- *Fuzzy Self-Tuning PID Controller* - where expert knowledge in the form of a fuzzy algorithm is used to tune the values of a PID controller for the control of an unknown plant [10].

- *Fuzzy Gain Scheduling Controller* - similar in principle to the classical gain scheduler only a fuzzy interpolation is implemented between the different linear controllers [66].

- *Fuzzy State Space Pole Placement Controller* - as in the case of Fuzzy Self-Tuning PID Controller, the state feedback gains are adjusted by a fuzzy algorithm [67]. This has been applied to hydraulic drives [68].

The literature found dealing with this area of adaptive fuzzy control was scant. No industrial applications of this methods were found although a commercial fuzzy self-tuning PID controller is available from the company OMRON.

2.10. Conclusion

This chapter has presented an overview of stand-alone adaptive fuzzy control strategies with emphasis placed on on-line adaptive methods. As in classical adaptive

control two main classes were found to be prevalent ; the direct and indirect adaptive fuzzy control methods.

Among the direct adaptive fuzzy control methods, the self-organising controller is the most common. This method allows the direct on-line adaptation of the consequent values of a fuzzy rulebase. Based on this prevalence, this control strategy was chosen for further evaluation. The investigation into the capabilities of the SOC is detailed in Chapter 5 of this thesis.

The indirect adaptive fuzzy control literature was of a high quality. The reported results for the single step fuzzy controllers were promising and as this is a model based approach, it is extendable to multi-variable control. Based on this extendability, the single step predictive fuzzy controller was chosen for further evaluation. The investigation into this method is also contained in Chapter 5. As this control strategy utilises an adaptive fuzzy model of the plant to be controlled, adaptive fuzzy models of the warm water process were developed. The structure, learning method and modelling accuracy of these fuzzy models are reported on in Section 4.9.

Chapter 3 - *Warm Water Process Hardware and Software*

3.1. Introduction

3.1.1. General Introduction

This chapter describes the warm water process plant, its associated hardware and the data acquisition hardware and software used for interfacing the plant to a computer. The hardware and software used for this research can be divided into three categories:

- the *warm water process plant* with its associated actuators and sensors,
- the *signal conditioning amplifiers and computer with its analogue to digital (ADC) and digital to analogue (DAC) converters* and
- the *ADC/DAC interface control software.*

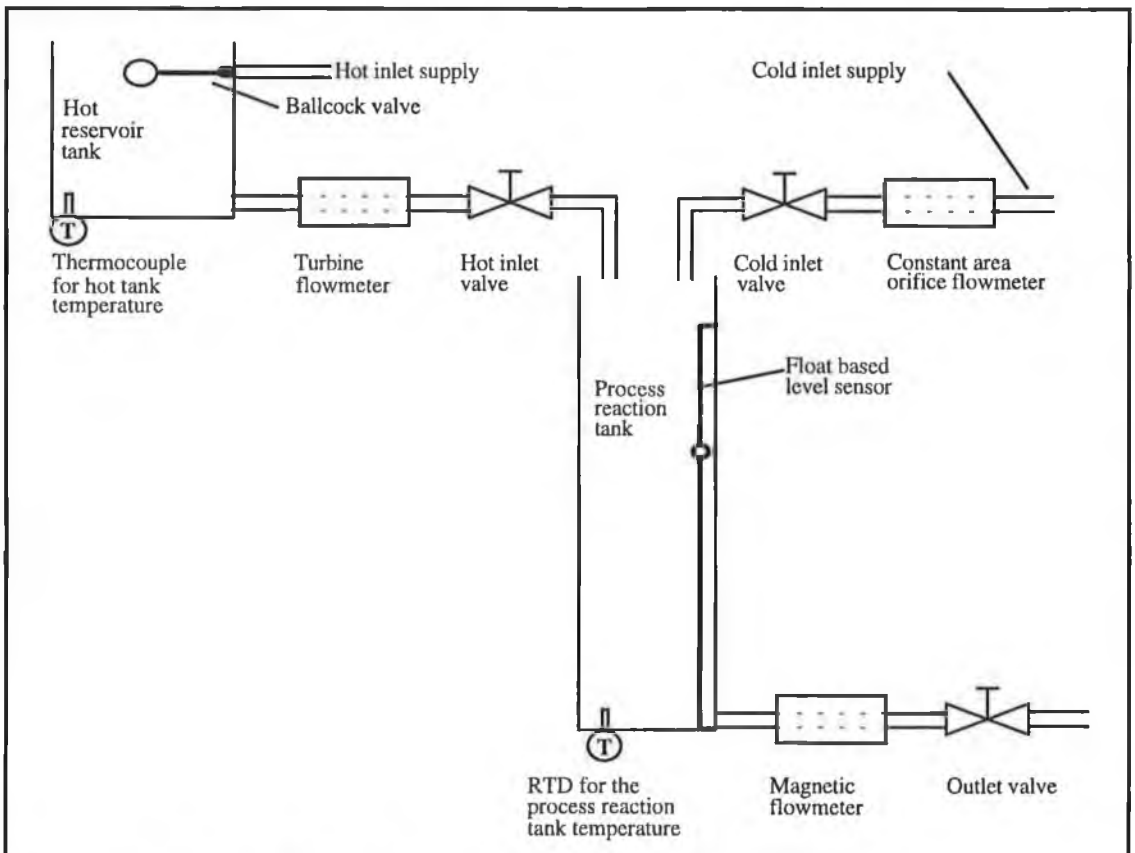


Figure 3.1 - Physical block diagram of the warm water process plant.

3.1.2. Overview of Chapter Structure

Section 3.2. presents a comprehensive description of the warm water process including its sensors and actuators. The computer and the interface cards utilised for

the control of the plant are detailed in *Section 3.3.* In order to interface the plants sensors and actuators to the ADC and DAC cards, a signal conditioning circuit was designed and constructed. The design and specification of this signal conditioning circuit is discussed in *Section 3.4.* The interrupt driven software structure utilised for data acquisition and control is described in *Section 3.5.* Figures 3.11, 3.12, 3.13, 3.14, 3.15, 3.16 and 3.17 on pages 55 to 58 show colour photographs of some of the warm water process components.

3.2. The Warm Water Process Plant

3.2.1. Physical Description

The warm water process plant consists of *two fibreglass tanks* - the hot reservoir tank and the process reaction tank - t*hree actuators* and *six sensors.* All pipes used within the plant are copper, with a diameter of 0.5 inchs. Figure 3.1 (see page 37) contains a block diagram of the plant and Figure 3.2 shows a schematic diagram of the plant.

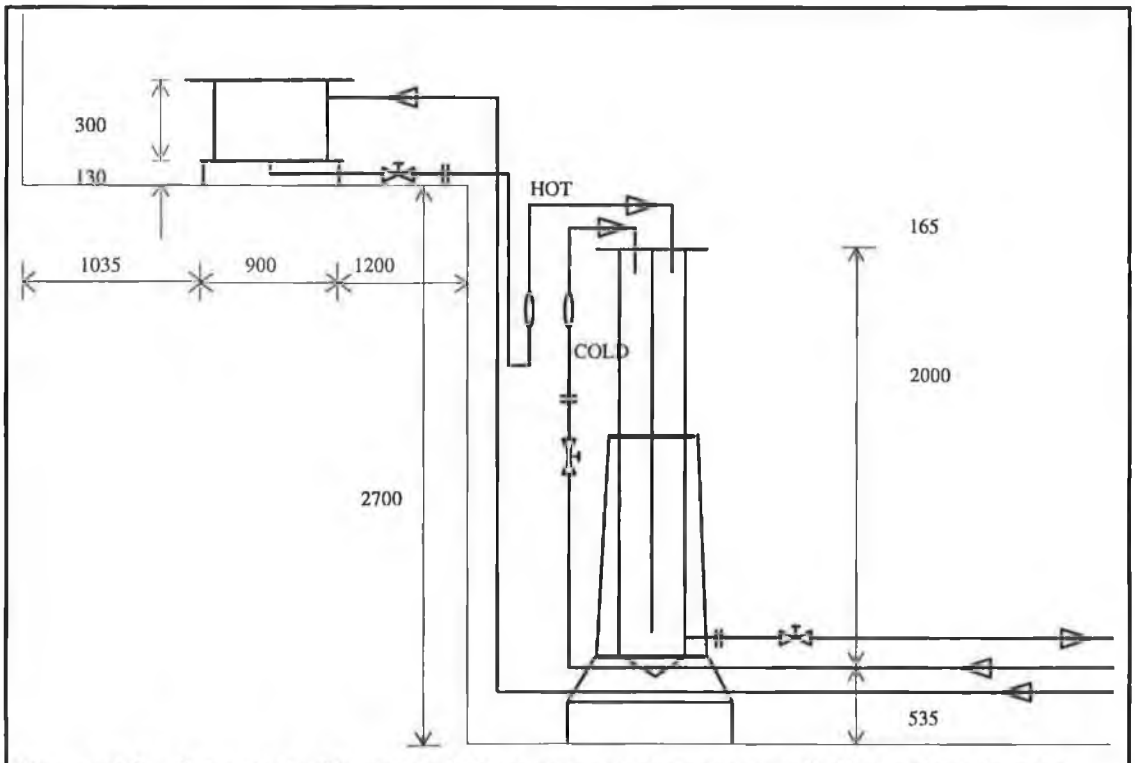


Figure 3.2 - Schematic diagram of the warm water process - all dimensions in mm.

The *hot reservoir tank* is rectangular in shape measuring 80 cm by 80 cm by 60 cm and serves as a reservoir for hot water. In order to achieve a constant pressure head of hot water for the process reaction tank, the hot reservoir tank was found to be necessary. The level of hot water in the hot reservoir tank is kept constant by a ballcock valve on the inlet, thus ensuring a constant pressure head at the hot reservoir

tank outlet. In addition, the hot reservoir tank also contains a thermocouple to measure the temperature of the hot water. Figure 3.4 contains a schematic diagram of the hot reservoir tank.

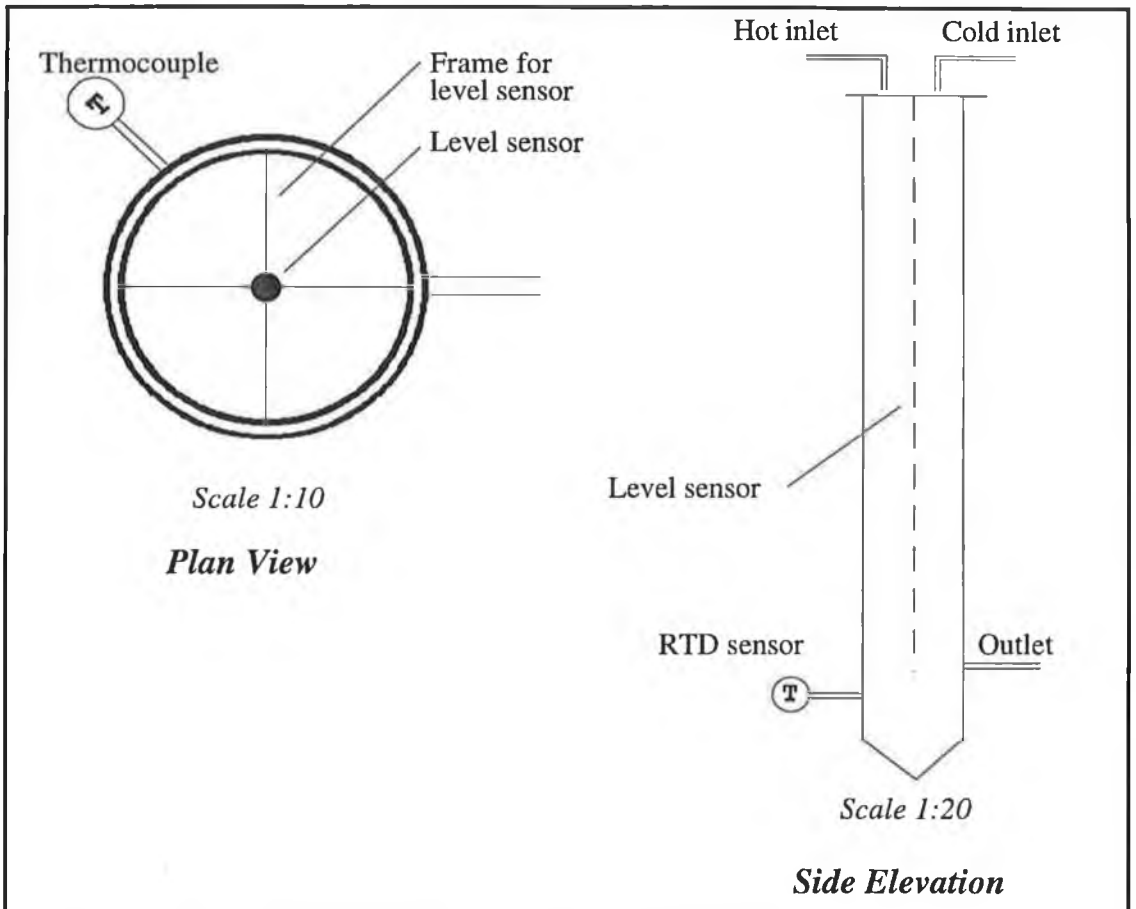


Figure 3.3 - Schematic diagram of the process reaction tank.

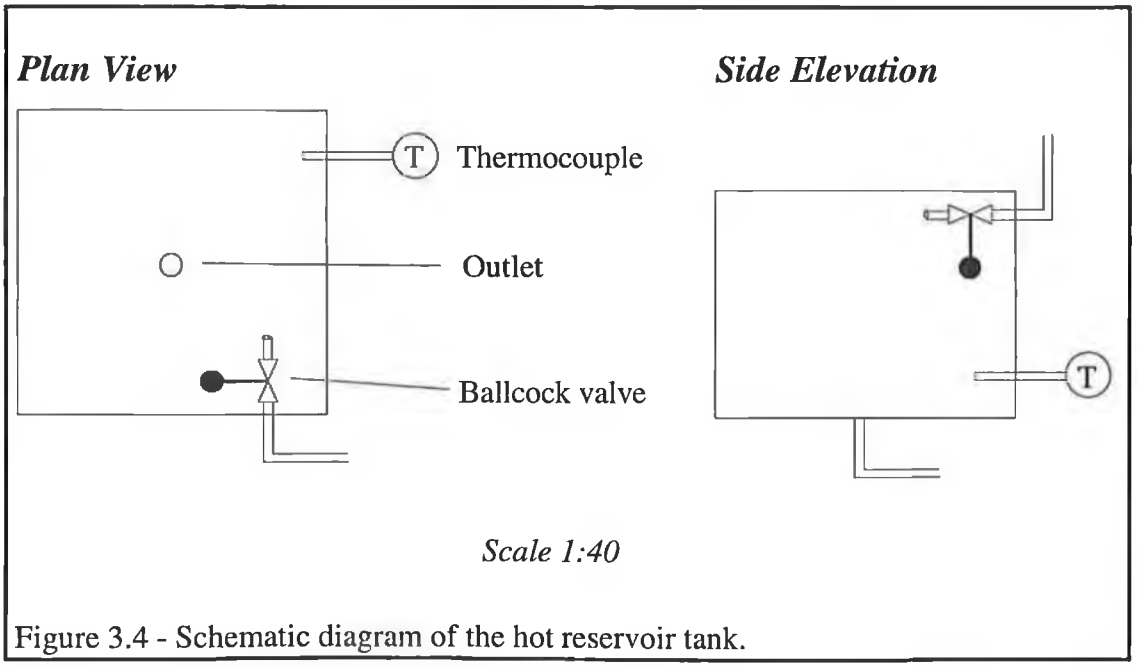


Figure 3.4 - Schematic diagram of the hot reservoir tank.

The *process reaction tank* is of a cylindrical construction with a height of 180 cm and an inner radius of 17.4 cm. This tank has an inlet for both hot and cold water, each of which is situated freely over the top of the tank. The outlet is situated approximately 10cm above the bottom of the tank. Figure 3.3 (see page 39) contains an exact graphical description of the process reaction tank. The process reaction tank contains a level sensor and a Resistance Thermometer Device (RTD) as a temperature sensor. Although normally standard for any mixing tank, the process reaction tank possesses no device to ensure proper mixing of the two inlet flows. Due to this lack of a mixing device and the long cylindrical geometry, temperature gradients along the length of the tank are to be expected.

3.2.2. Actuators

The actuators utilised in the warm water process are the *cold inlet valve*, the *hot inlet valve* and *outlet valve*. All three of these valves are pneumatically powered with the actuating signal originating from a 4-20 mA current loop.

The valve used for control of the *cold inlet flow* is the Masoneilan Varipak 28000 series. As this valve is pneumatically powered, a Foxboro current to pneumatic converter - model E69F-BI2 is is also used [69]. One of the more interesting features of this pneumatically controlled valve is the facility to adjust the value of the valve discharge co-efficient Cv [70].

For control of the *hot inlet flow*, a pneumatically controllable valve from Sensycon 23/16 series has been combined with a Sensycon electropneumatic valve positioner - type 23/55-21 [71,72]. The valve has exchangeable plug and seal, the option of an electric motor actuator, the choice between linear and equal percentage characteristics and air to open or air to close operation.

The *outlet flow* is controlled by a Masoneilan Camflex II Series 35002 valve together with a Masoneilan series 4600 pneumatic positioner. The valve is characterised by its eccentrically rotating plug. The series 4600 pneumatic positioner allows the choice between air to open and air to close, as well as linear, split linear or equal percentage characteristics [73,74].

3.2.3. Sensors

The warm water plant possesses a total of eight sensors.

Two of these are for temperature measurement :

1. the temperature sensor in the hot reservoir tank and

2. the temperature sensor in the process reaction tank.

There are five sensors for flow measurement :

3. the flowmeter in the hot inlet line,

4. the flowmeter on the cold inlet line,

5. the flowmeter in the outlet line and

6., 7. the variable area flowmeters in both hot and cold inlet lines.

One sensor for level measurement is also utilised :

8. the level sensor in the process reaction tank.

Apart from the two variable area flowmeters, all sensors transmit their corresponding signals by means of 4-20 mA current loops. These loops offer the advantage of a very high signal to noise ratio as the signal is proportional to a current, but necessitate signal conditioning to convert the signal from a current to a voltage to enable further processing by an analogue to digital converter. The six plant sensors are described in Sub-Sections 3.2.3.1 - 6.

3.2.3.1. Temperature Sensors

Both temperature sensors, one installed in the hot reservoir tank and the other situated in the process reaction tank, are manufactured by Bush Beach Engineering. The hot reservoir utilises a *Pt100 RTD* whereas the process reaction tank contains a *type K thermocouple*, both sensors have ranges of 0° to 100° Celcius. The two sensors consist of a connection head, a head mounted 4-20 mA transmitter and stainless steel jacketed probes in fabricated stainless steel pockets. The immersion length of both probes is 15 cm [75]. An external voltage source in the current loop in order to power the 4-20 mA amplifers is required for both devices.

3.2.3.2. Variable Area Flowmeters

In order to give some visual indication of the hot and cold inlet flows, both the hot and cold inlet lines contain variable area flowmeters. These flowmeters are manufactured by Perflow Instruments Ltd and are both from the *MAJOR range, type FR6S* with a measurement range of 0.03-0.36 L/sec. [76].

3.2.3.3. Hot Inlet Flowmeter

The hot inlet flowmeter is a *turbine flowmeter - model 1/2-82T4E4* from Foxboro combined with an analogue amplifier of type PA420. A measurement range of 4.5 - 45 L/minute is typical for this device. Some further constructional specifications are: ball-sleeve bearing, a 17-4 PH stainless steel rotor and a tungsten carbide rotor shaft [77]. A 4-20 mA current signal proportional to the signal from the turbine flowmeter is transmitted by the amplifier [78]. As in the case of the temperature sensors, an external voltage source must be placed in the current loop in order to power the 4-20 mA transmitter.

3.2.3.4. Cold Inlet Flowmeter

The concept of the Venturi meter [79] is used to measure flow on the cold inlet. The orifice in the cold inlet supply is constructed from 3 mm thick 316 stainless steel and the orifice characteristic is in accordance with BS1042/ISO 5167 [80]. Fluid from both sides of this orifice is fed to a differential pressure measuring device - *843 DP-B2I1SS Cell TRANSMITTER* from Foxboro. This transmitter has low/high span ranges of 0-25 and 0-100 psi, an upper range limit of 100 psi [81] and requires an external voltage source within the 4-20 mA current loop for operation.

3.2.3.5. Process Reaction Tank Level Sensor

The level sensor contained in the process reaction tank functions through the use of a magnet-operated float switch. This sensor consists of a long tube which spans the length of tank and contains a set of resistors in series, each of which can be connected to ground through a magnetic switch. A float containing a magnet and which can move freely along the tube swims atop the fluid in the tank and closes the magnetic switch in its vicinity. Thus, using the potentiometer method, the voltage dropped across the resistance between the top of the sensor tube and the float is proportional to the level of fluid in the tank. The signal from this level sensor is not continuous but discrete in nature with a resolution of two centimetres.

KSR Kuebler Control Engineering Limited manufactures the level sensor utilised in the process reaction tank [82]. The main constituents of the sensor are a two metre length of *AEV2-VK12-L2000-SV tubing* and a *4-20 mA transmitter of type MUA*. In order to power the transmitter an external single rail voltage of between 15 and 30 volts must be connected to the amplifier supply terminals [83]

3.2.3.6. Outlet Flowmeter

For measurement of the outlet flow a magnetic flowmeter is utilised. Magnetic flowmeters are suitable for the measurement of volumetric flow rate of electrically conductive liquids. The complete flowmeter consists of a Foxboro Magnetic Flow *Tube Model - 801H-WCR-AG* and a Foxboro magnetic flow *transmitter Model - 8000P-B13-G.* The transmitter uses a pulsed DC technique in order to excite the 8000 series flow tubes. The output signal of the transmitter can be either a 4-20 mA current or a pulsed output. The transmitter contains its own power supply and thus no external voltage source is necessary within the 4-20 mA current loop [84,85].

3.3. Computer Interface Circuitry

3.3.1. Description of Computer

The computer used for the control of the plant is an IBM compatible PC from Siemens-Nixdorf. This PC possesses an *Intel 80486DX2 50MHz microprocessor* which includes an *80387 mathematical co-processor on chip.* The system bus is constructed according to the *VESA Local Bus* specification. In addition to two VESA local bus slots there are *4 standard ISA bus expansion slots* which permit the addition of additional peripheral devices to the computer bus. In order to allow the interfacing of the plant transducers and actuators to the computer both *analogue to digital* and *digital to analogue converters* to fit the ISA expansion slots were purchased (see Sections 3.3.2. - 3.)

3.3.2. Analogue to Digital Converter

An analogue to digital converter (ADC) was necessary in order to allow the interfacing of the plant transducer signals to the computer. Specifications deemed suitable for the ADC card are listed below :

- *Number of channels* - at least six channels were required for full interfacing of all transducers.

- *Converter resolution* - 12 bits were deemed to be suitable. This corresponds to a voltage of 4.883 mV/bit for a bipolar input with a range of ±10 volts.

- *Interrupt facility* - data acquisition should be performed via a hardware interrupt. Through a system hardware interrupt, data acquisition can take place in the background, without any form of polling.

- *Self-timer* - the required hardware interrupt should be triggered by an on board programmable timer. This allows a simple specification, and if need be, the modification of the required sampling time for the controller algorithms.

- *software drivers* - in order to increase the ease of programming, software drivers for the ADC card in the C programming language were required.

The *PCL-812PG ADC* from the company Advantech with sixteen single-ended bipolar input channels, 12 bit resolution, interrupt facility, on board timers and software drivers fulfilled all of the above criteria [86].

3.3.3. Digital to Analogue Converter

In order to drive the plant actuators with the computer, a digital to analogue conversion card is required. The required criteria for the digital to analogue conversion card are detailed below:

- *Number of channels* - at least three analogue output channels were required for the plant.

- *Resolution* - 12 bits resolution.

- *4-20mA output capability* - all outputs should be in 4-20mA current loop form to minimise the need for further signal processing.

- *Software drivers* - to permit ease of programming software drivers in the C programming language.

The digital to analogue converter purchased was the *PCL-726 DAC card*, also from the company Advantech. This card has 6 output channels each with 4-20 mA current

loop capability and 12 bit resolution. Due to the simple register structure of the board, no software drivers were required [87].

3.4. Signal Processing Board

3.4.1. Introduction

This section describes the signal conditioning circuit that was developed in order to create the interface between the plants transducers and the analogue to digital conversion (ADC) board. As previously described, all transducer signals are transmitted by 4-20 mA current loops. The ADC possesses 16 single ended bipolar inputs which require a voltage as an input signal. The four main functions of the signal conditioning board can be summarised as follows :

- *current to voltage conversion,*
- *voltage amplification,*
- *anti-aliasing filtering and*
- *offset voltage compensation.*

The realisation of these four functions is described in detail in the following Sections 3.4.2. - 5. A complete schematic diagram of a single channel of the signal processing circuit is shown in Figure 3.5 (see page 47).

3.4.2. Current to Voltage Conversion

Current to voltage conversion of the 4-20 mA signal is achieved by placing a *resistor in series within the 4-20 mA current loop.* The voltage dropped across this resistor is given by equation (3.1).

$$V = I_{signal} R \qquad \text{where } R = R1 + RV1 \tag{3.1}$$

The values of R1 and RV1 should be chosen in conjunction with the values for the load impedances specified by the different sensor amplifiers. A value of 470 Ω for R1 was chosen for all current loops [75,78,81,83,84]. The overall gain of signal conditioning board was then specified so that the input range of the ADC is matched by that of the output voltage of the signal processing board. The value of *unity*

chosen for the overall gain of the complete circuit in conjunction with a 500 Ω series resistor in the 4-20 mA current loops gave rise to a voltage range of 2.5-12.5 volts for a 4-20 mA current input signal. This range is converted to 0-10 volts by the second amplifier stage. The variable resistor RV1 was placed in series with R1 so that errors due to tolerances in R1 could be adjusted to a minimum.

3.4.3. Voltage Amplification

The circuit consists of *two amplification stages*.

The *first stage* is a single instrumentation amplifier based around the operational amplifier A1 [88,89]. The gain of this stage, $Gain_{A1}$, neglecting C1 is given by equation (3.2).

$$V_{out} = \frac{R3}{R3+R5}\left(1+\frac{R4}{R2}\right)V_2 - \frac{R4}{R2}V_1 \quad \text{where } V_{in} = V_2 - V_1$$

$$Gain_{A1} = \frac{R4}{R5} \qquad \text{where R2 = R3 and R4 = R5} \qquad (3.2)$$

The gain chosen for this instrumentation amplifier was 0.45. This value helps to maintain the linearity of the stage as the output voltage does not approach the supply voltage. The capacitor C1 contained in the negative feedback loop reduces the high frequency gain of the stage and thus improves noise rejection [89,90].

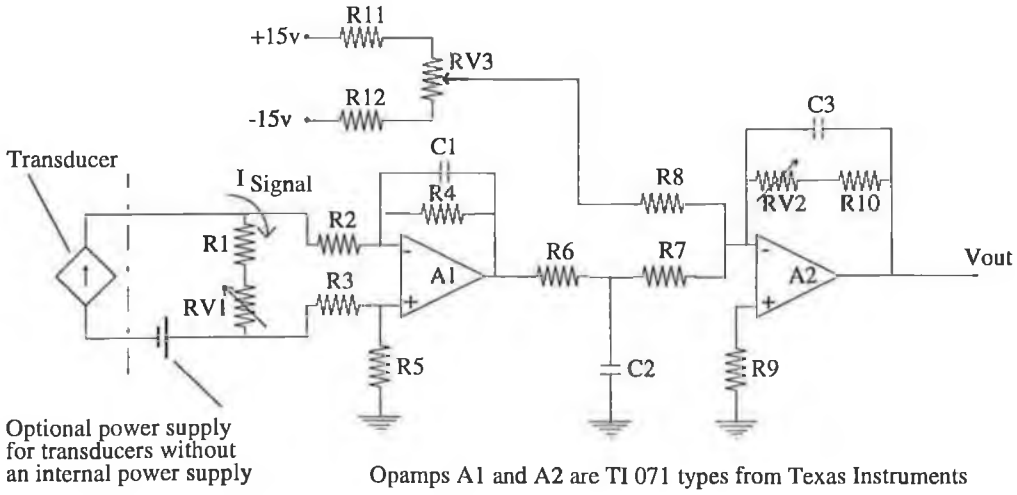


Figure 3.5 - Schematic diagram of a single channel from the signal conditioning board.

The *second stage* in the signal conditioning circuit is an inverting summing amplifier built around operational amplifier A2 [89]. This amplifier stage possesses two

summing inputs with seperately definable gains. The overal gain of this stage is given by equation (3.3) and has a value of 2.75.

$$Gain_{A2} = \frac{R10 + RV2}{R7} \tag{3.3}$$

This results in an output voltage range of 2.5-12.5 volts. The variable resistor RV2 sited in the negative feedback loop of A2 allows the reduction of any gain errors resulting from resistor tolerances in both amplification stages. As in the instrumentation amplifier, the capacitor C3 in the feedback loop reduces the high frequency gain of the stage helping to ensure stability. The resistor R9 is inserted in the non-inverting input to ground and serves to reduce the effects of bias current drift [89,90]. Its value is given by equation (3.4).

$$R9 = \frac{R8 \cdot R7 \cdot (RV2 + R10)}{R8 + R7 + (RV2 + R10)} \tag{3.4}$$

3.4.4. Anti-Aliasing Filtering

A simple first order passive anti-aliasing filter based around the RC pair R6 and C2 has been inserted between the two amplifier stages. This configuration offers a good compromise between circuit complexity and performance. The cut-off frequency f_c is given by (3.5).

$$f_c = \frac{1}{2 \cdot \pi \cdot R6 \cdot C2} Hz \tag{3.5}$$

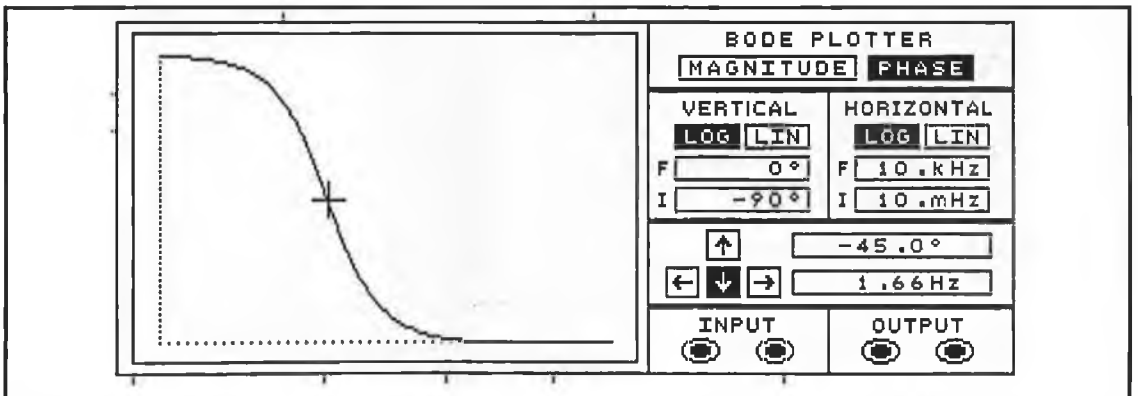


Figure 3.6 - Simulated Bode plot of one channel of the signal conditioning circuit.

The value of R6 should be kept low with respect to R7 as it otherwise contributes to the gain of second stage. Figure 3.6 shows the simulated phase response of one channel of the signal conditioning circuit. This Bode plot was calculated by the circuit

simulator "Electronic Work Bench". The response is first order with a crossover frequency of 1.66 Hz - the cross hairs are positioned on this point. With the resistor R6 equal to 47 kΩ and C2 equal to 2.2 μF, the theoretical crossover frequency calculated using equation (3.5) is 1.539 Hz. The simulated crossover frequency, as described above, is 1.66 Hz. Thus the anti-aliasing filter functions as expected when positioned between the two amplifier stages. The crossover frequencies for each of the six channels are detailed in Appendix A.

3.4.5. Offset Voltage Compensation

Because the transducers transmit their signals over 4-20 mA current loops the output voltage range of the signal processing circuit is 2.5-12.5 volts. By introducing 2.5 volts at the other summing inverting input of A2 this range is shifted to 0-10 volts. This offset voltage is provided by the potential divider consisting of resistors R11, R12 and RV3. RV3 allows the fine adjustment of the offset voltage within the limits set by R11 and R12. These limits, $V_{OffsetHigh}$ and $V_{OffsetLow}$, are expressed in equation (3.6) and (3.7).

$$V_{OffsetHigh} = \frac{Vs \cdot R12}{R11 + R12 + RV3} \tag{3.6}$$

$$V_{OffsetLow} = \frac{Vs \cdot R11}{R11 + R12 + RV3} \tag{3.7}$$

3.4.6. Circuit Simulation, Construction and Calibration

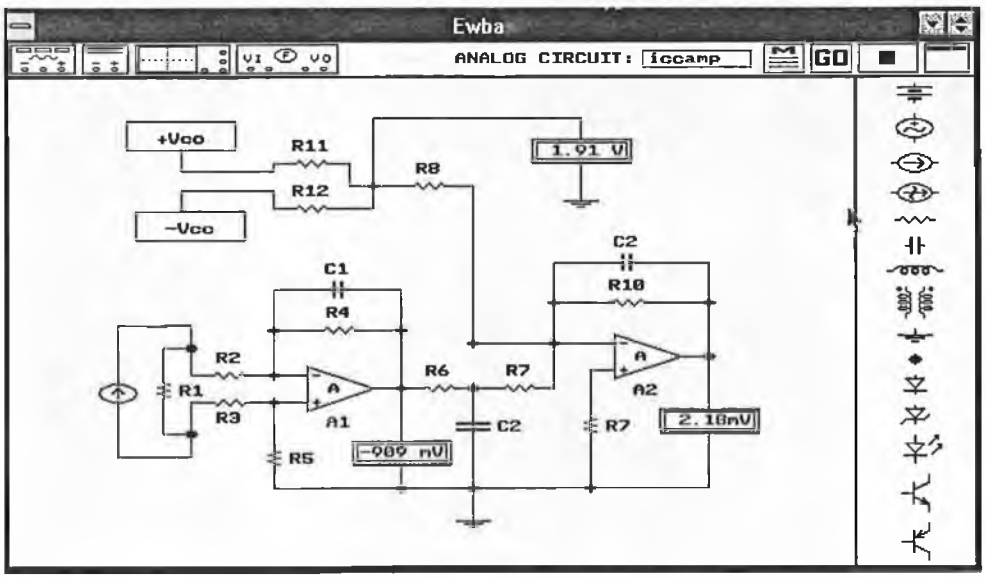


Figure 3.7 - Electronic Workbench simulation of the signal processing circuit.

All initial designs of the circuit were simulated using the software package *Electronic Workbench*, see Figure 3.7 above. Following successful simulation the prototype of

the signal conditioning board was constructed on Vero-Board. Using a high precision multimeter - Hewlett Packard 3478A - the current conversion resistors for all six channels were adjusted to a value of 500 Ω. Using a constant voltage source the gains and offset voltages of all six channels were altered to give the required output voltage range of 0-10 volts. All component values and specifications are contained in Appendix A.

3.4.7. Power Supply Circuits

Power supply circuits were designed and built for the signal conditioning amplifier, for the level meter amplifier, see Section 3.2.3.5, and for the four transducer current loops that required an external series supply voltage. To power the level meter and the four current loops, a single rail DC voltage of 24 volts was required. For the signal conditioning circuit a dual rail supply of ±15 volts was necessary. Figure 3.8 (see page 50) shows the schematic diagrams for both power supply circuits.

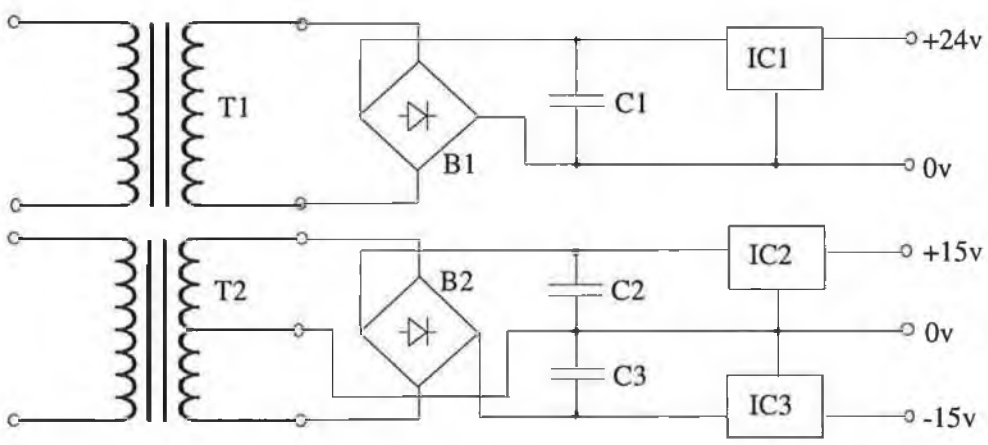


Figure 3.8 - Schematic diagrams of the two power supply circuits.

The component types and specifications for both power supply circuits are listed in Appendix A.

3.5. Interrupt Driven Interface Software

Having designed and built the signal conditioning board, software was required to enable data acquisition, ADC and DAC card control and the investigation of closed loop control strategies for the warm water process. The basic requirement for this software was that the hot, cold and outlet valves could be controlled independently. The time constants of these valves are of the order of 0.5 seconds, where as those of the flow and temperature variables for the process reaction tank are of the order of

500 seconds. Due to this large difference in the plant time constants, a *multi sampling rate controller structure* was deemed to be necessary.

There are two common approaches to data acquisition and control software structure - *data polling* and an *interrupt service routine.*

The *data polling approach* is the simpler option. The main program is run and when finished a timing loop waits for the completion of sampling, at which point the data is logged and the main program restarts. The main disadvantages associated with this method are the fact that the micro-processor must actively wait and that the execution time of the main program must be shorter than the sampling period.

The *interrupt service routine approach*, whilst requiring more complicated software, is a more elegant technique. An external source triggers an interrupt service routine (ISR) function which can be user defined. This ISR can, for example, be designed to perform data acquisition and control. The ISR technique eradicates the waiting intrinsic to data polling and allows the main program to have effectively any temporal length, providing the ISR execution time is sampling period. The difference between the polling approach and the interupt service routine is depicted in Figure 3.9.

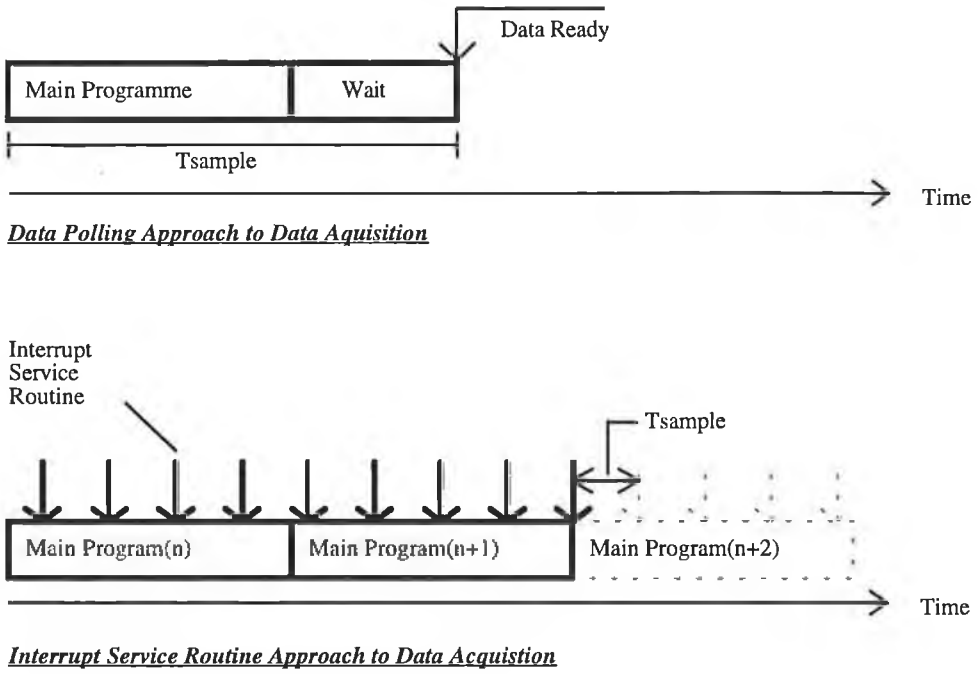


Figure 3.9 - Interrupt driven software and the data polling approachs to data acquisition.

Based on these considerations, an interrupt driven system was adopted for the data acquisition and control requirements of the warm water process. This choice allows background control of the two inlet valves and the outlet valve whilst control of the

process reaction tank and other functions such as graphics and user input can be run in the foreground.

The Advantech PCL 812PG ADC card possesses integrated timer/counter functions, provided by the on board Intel 8253-5 timer-counter chip. The 8253-5 has three user programmable 16-bit timer/counters each with six different modes. More detailed information about the Intel 8253-5 is contained in [91]. Counters 1 and 2 in the PCL-812PG ADC have been cascaded together and a quartz crystal clock, with an oscillation frequency of 2 MHz, serves as an input to counter 2. Thus, the user can program the cascaded 16 bit counters 1 and 2 to provide a rate generator with a period of between 35.79 minutes and 500 nano-seconds. This rate generator can be then used to trigger one of the PC interrupt request lines, i.e. IRQ2 to IRQ7. The counters can be programmed either by direct register addressing or through the software drivers provided with the ADC card.

To enable an external interrupt on the IBM PC and to install a user defined interrupt service routine, several steps are necessary. Detailed instructions and explanations for this procedure can be found in [91,92].

For data acquisition and control of the three valves, a sampling time of 0.05 seconds was implemented. The procedure used for the installation of the interrupt service routine to be triggered on IRQ7 by the Advantech ADC card is described by the following :

- Set jumper on ADC card for IRQ7.

- Program the 16 bit counters 1 and 2 with 100_{10} and 1000_{10} respectively in rate generator mode (Counter mode 2) using direct register addressing, this sets the desired sampling time of 50 milli-seconds.

- Write the Interrupt Service Routine function and declare the function to be an interrupt using the ANSI C keyword *interrupt:*

```
void interrupt FunctionName(void)
{....
}
```

- Disable all PC interrupts with the ANSI C function *disable().*

- Save the current IMR register value (Address Hex 21) and unmask (set to zero) the IRQ7 interrupt flag in the IMR, thus enabling the IRQ7 interrupt request line.

- Save the old interrupt vector for IRQ7 and replace it with the user defined interrupt service routine using the ANSI C function *setvect()* .

- Reenable all PC interrupts with the ANSI C function *enable()*.

The user defined interrupt service routine will now be called when an interrupt request is detected on the IRQ7 line. After the program has terminated, the old values of the IMR and the IRQ7 interrupt vector should be restored in order to ensure correct functioning of the PC when using other programmes. Two functions were written to install and remove the user defined ISR. The source code for these functions is contained in Appendix B.

During program execution some general system housekeeping is necessary in order to maintain correct functioning of the (ADC card driven) interrupt service routine. Figure 3.10 (see page 54) shows a flow chart representation of one ISR cycle. The house keeping commands are explained as follows :

- *Reset ADC Card interrupt request register* - after requesting an interrupt this register must be reset otherwise the IRQx line remains high allowing further ISR instances to be created.

- *Re-enable ADC card interrupt mode* - this is the triggering mode of the ADC card and must be set to interrupt mode after each ISR.

- *Acknowledge interrupt to PC* - this acknowledges the interrupt and allows further interrupt processing to proceed.

In order to test the Interrupt Service Routine structure, the sampling time was set to one milli-second and the program was run for six hours. Thus after over two million ISR calls the software was deemed to be reliable.

Finally, it should be mentioned that all the ANSI C software should be programmed with the compiler option for register optimisation set to *NONE*. Otherwise erroneous operation of the ISR structure with eventual seizing of the PC system was observed to occur. Further explanation of this option and all of the mentioned ANSI C functions is contained in [92,93,94].

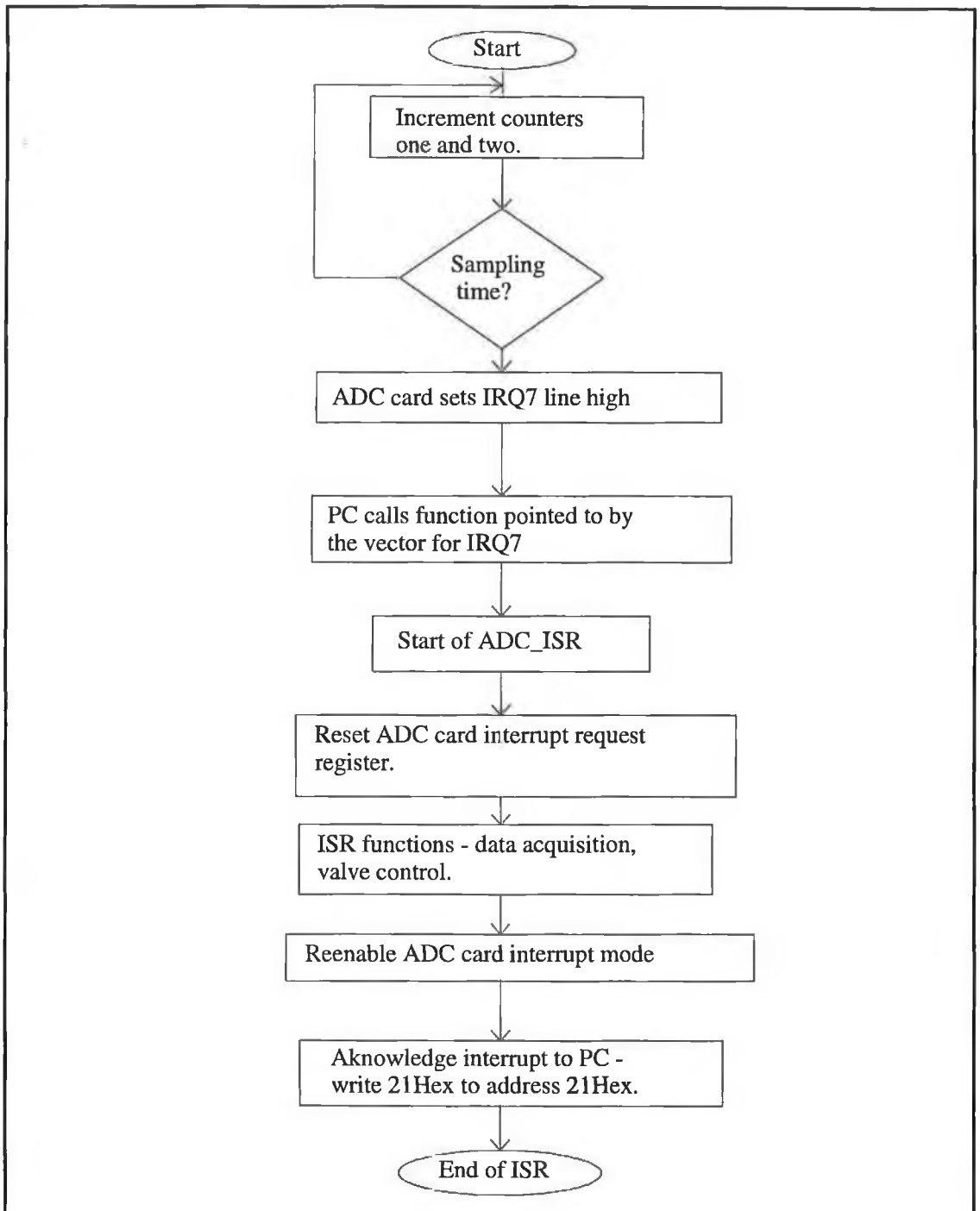


Figure 3.10 - Flow chart illustration of one interrupt service routine cycle.

3.6. Conclusion

This chapter has presented a detailed description of the constituent components of the warm water process. The design and construction of the necessary hardware and software for the interfacing to and control by a computer have also been detailed. Having now completed the necessary steps to allow data acquisition and control, the next chapter in this thesis is concerned with the modelling of the warm water process.

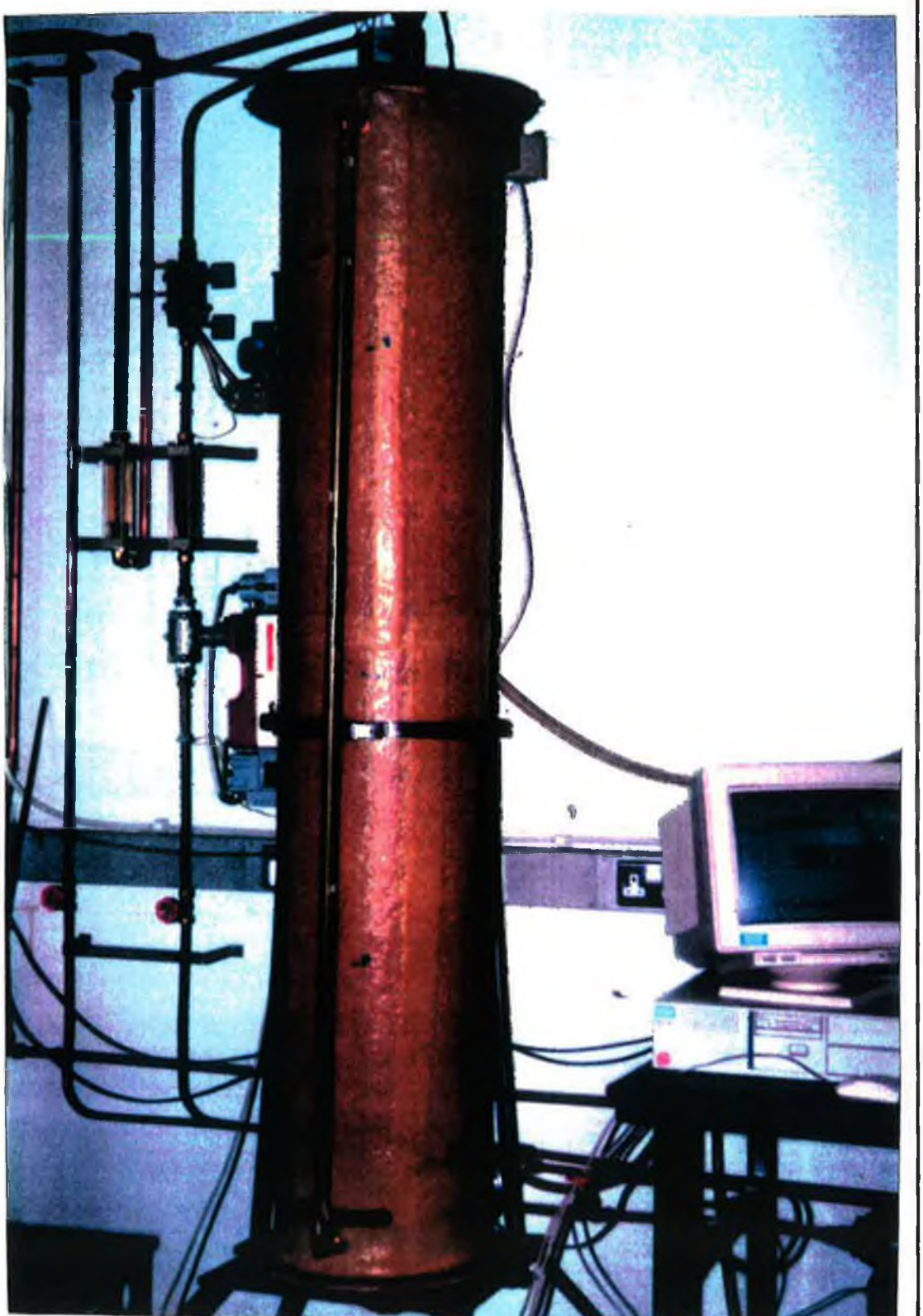


Figure 3.11 - Process reaction tank with control computer.

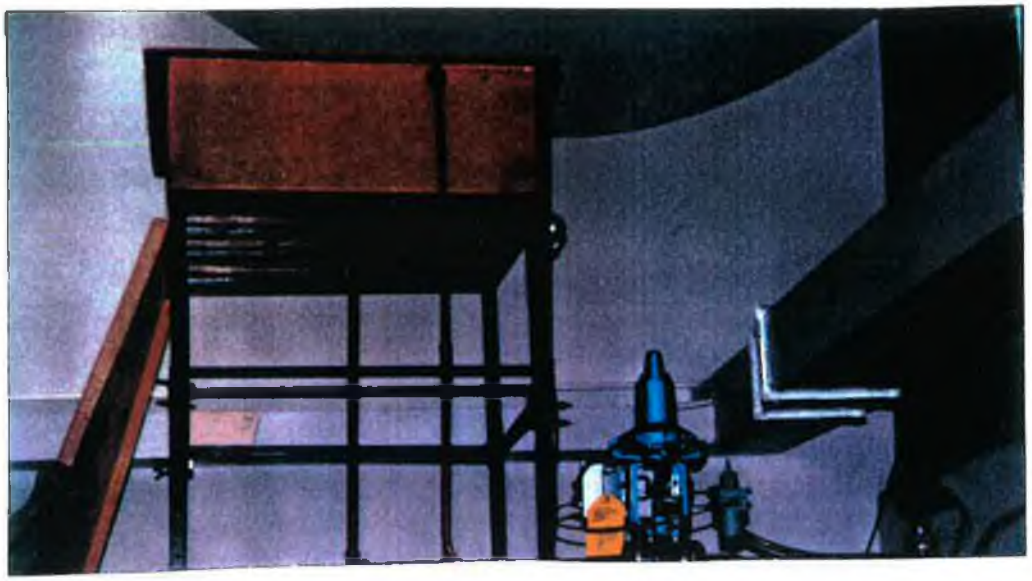


Figure 3.12 - Hot reservoir tank with hot inlet valve (left) and flowmeter (extreme left).

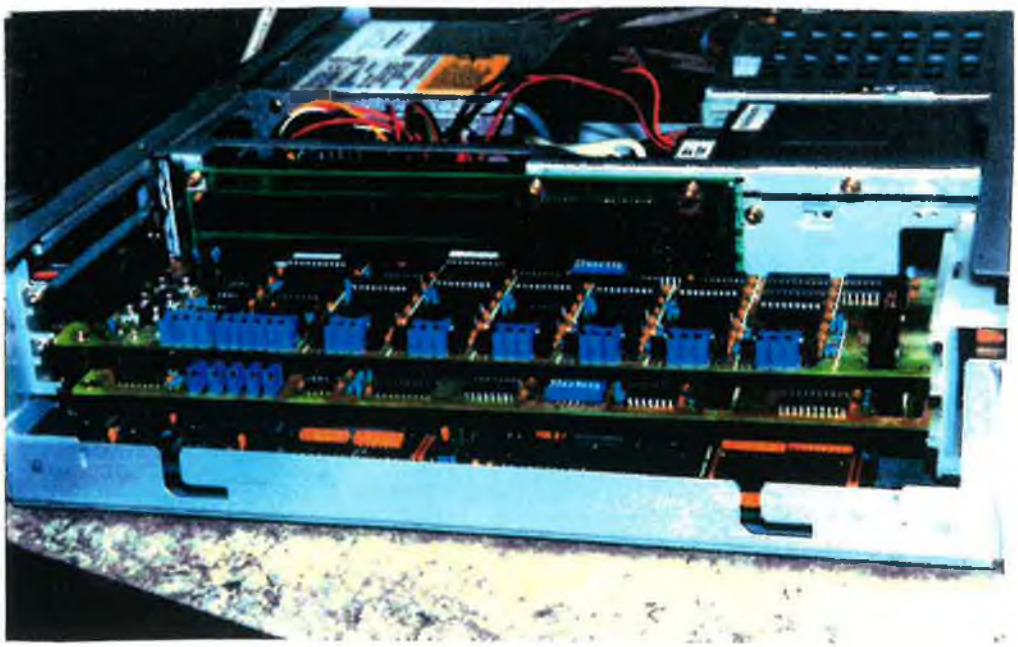


Figure 3.13 - Control computer with the ADC (lower) and DAC (upper) cards from Advantech.

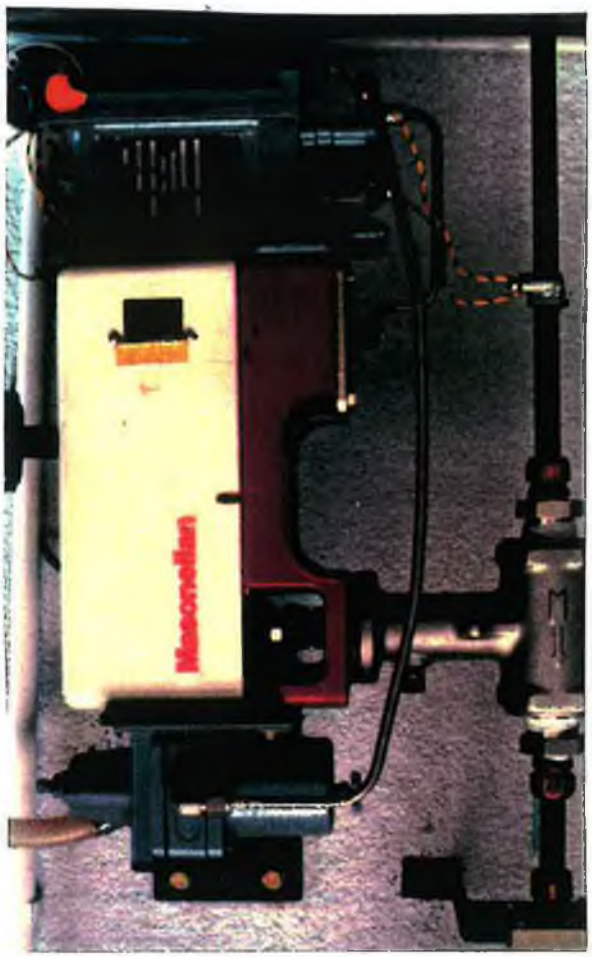


Figure 3.14 - Cold inlet valve.

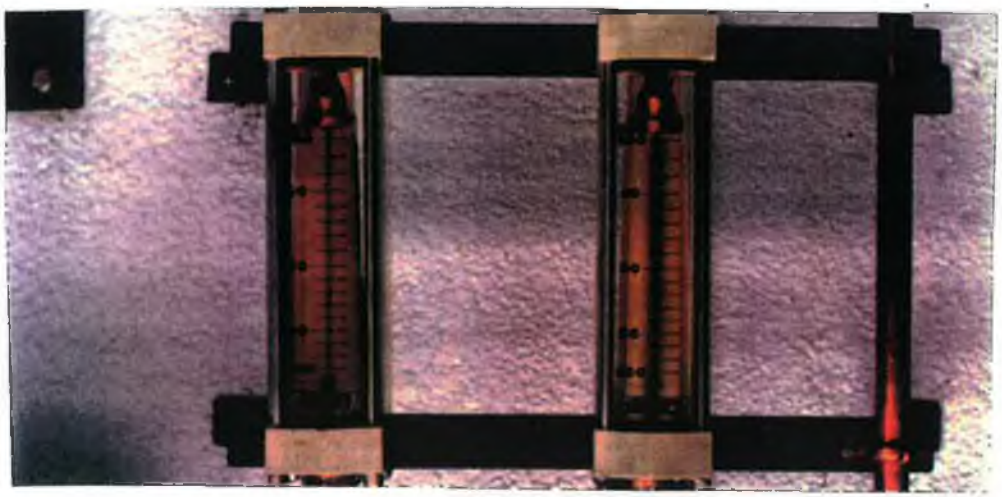


Figure 3.15 - Visual flowmeters from Perflow.

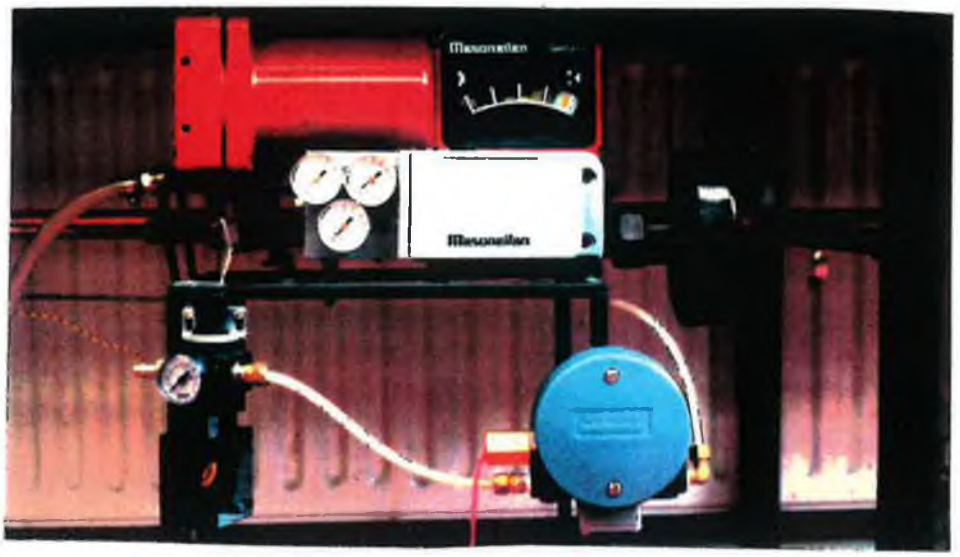


Figure 3.16 - Outlet flow control valve from Masoneilan.

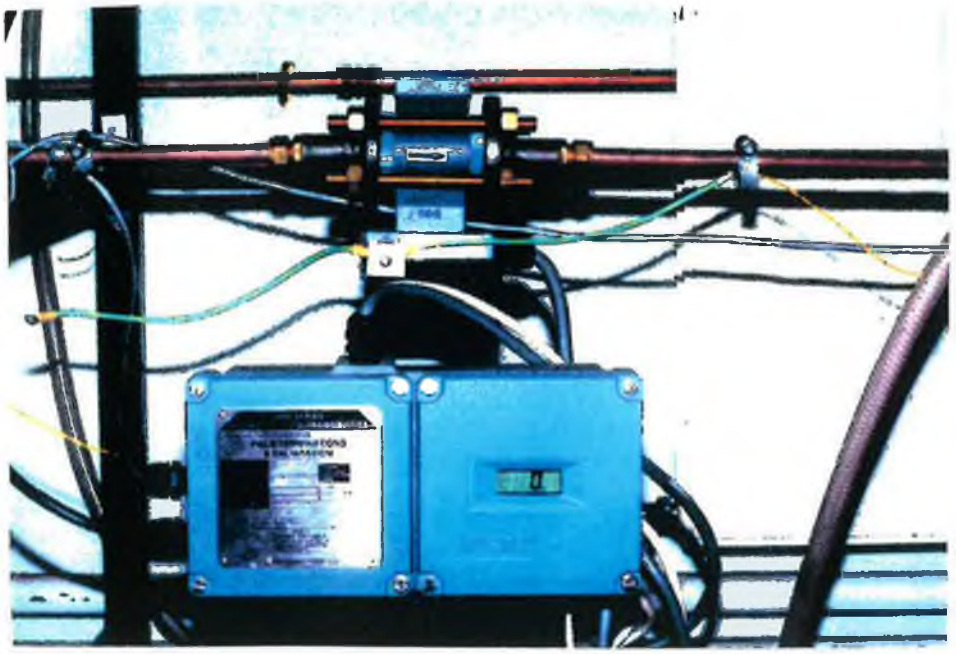


Figure 3.17 - Outlet flowmeter from Foxboro.

Chapter 4 - *Warm Water Process Modelling*

4.1. Introduction

4.1.1. General Introduction

In order to facilitate the design and simulation of a controller for a given plant, some form of mathematical model of the plant is required. The warm water process being a multivariable real world system represents a challenging modelling task. The mass flow of the system is characterised by a predictable non-linear response. The thermal energy behaviour of the process is strongly non-linear. This is due to the combination of the lack of a mixing device and the long and narrow geometry of the process reaction tank. In addition, the thermal energy response is strongly coupled with the mass flow of the system - being dependent on the hot and cold inlet flows. Four modelling options were applied to the warm water process - this chapter presents these strategies and their corresponding results. These different approaches allow diverse characteristics of the process to be examined, resulting in a better overall model of the plant.

The first option for modelling a given plant is the *derivation of a system model from physical first principles.* Through this derivation the structures of the model equations are obtained and the associated parameters are then acquired either through derivation or determination from suitable experiments on the plant.

Vector Mapping Methods (VMM) or *connectionist methods* represent a second option. The VMM associates output data with input data and thus learns a function from its input and output data without *a priori* knowledge of the function. One example is the *Artificial Neural Network* (ANN) of which one commonly used architecture is the feedforward *Multilayer Perceptron* (MLP). It has been shown that a three layer MLP can model a non-linear function to an arbitrary degree of accuracy [95]. During training the MLP is presented with both input and output data, and through the utilisation of a learning algorithm - often a modified gradient descent method such as the *backpropagation* algorithm - the connecting weights of the network are adapted to reduce the mapping error for the complete set of input and output data vectors. The user must choose the input and output vectors, the network structure and the learning method. When using an ANN for plant modelling, the data set used for training the neural network should contain state information from as much of the plant state space as possible, thus helping to guarantee the generality of the ANN model. As the science of neural networks is relatively new, no general theory exists to assist the user with the choice of the these network structures and

parameters, thus intuition and experience play key roles in successful application of this modelling technology.

A third option and a more recently developed modelling choice is the *fuzzy model.* As in the case of the ANN model, the fuzzy model maps an output data vector to an input data vector and has been termed a *fuzzy associative memory* [96]. The inputs to the model, the number and shape of membership sets for each variable, the logical operators, the storage method for the rulebase, the learning algorithm and the defuzzification method all need to be specified by the user.

The fourth option for modelling of the warm water process is that of *linear system identification.* This involves the determination of both the structure and parameters for a model through tests on the plant. One wide spread example of a model structure is that of a second order model with dead time. The open loop gain, natural frequency, damping coefficient and dead time coefficient can be determined by exciting the system with, for example, a combination of white noise and a square wave. This method is applied indirectly to the warm water process - through identification of the ARX models of the ANN model of the warm water process in simulation. This method, although applied indirectly, helps to gain insight into the dynamics of the process.

4.1.2. Overview of Chapter Structure

The following *Section 4.2.* of this chapter concerns itself with the calibration of the flowmeters of the warm water process. The design of PI controllers for the control of the flow through the hot and cold inlets is contained in *Section 4.3.* Following this, the method used for data acquisition from the warm water process is presented in *Section 4.4.* The derivation of a warm water process model from physical first principles is detailed in *Section 4.5.* Development and evaluation of an artificial neural network (ANN) model of the warm water process is outlined in *Section 4.6.* The MATLAB simulation of this ANN model of the warm water process is detailed in *Section 4.7.* The description of the development of linear first order ARX models of the warm water process around an operating point is found in *Section 4.8.* The development of a strategy for on-line adaptive fuzzy modelling, termed *supervised adaptive fuzzy modelling*, is elaborated upon and analysed in *Section 4.9.* This development is performed by analysing different types of fuzzy models for a simple first order system. The use of the developed adaptive fuzzy modelling approach in developing separate fuzzy models of the mass flow and thermal behaviour of the warm water process and the corresponding modelling results are detailed in *Section*

4.10. The last *Section 4.11.* summarises the results from the four modelling methods and concludes the chapter.

4.2. Flowmeter Calibration

This section describes the tests undertaken to calibrate the inlet and outlet flowmeters. As *initial* characteristics, a first order polynomial function was used for both the outlet and hot inlet flowmeters. This is due to the fact that both turbine and magnetic flowmeters have a linear relationship between the actual flow and flowmeter output signal. A second order polynomial function was utilised for the cold inlet flowmeter, as the pressure drop across an orifice with respect to the flow follows a squared law function [97].

The determination of the exact characteristics of the hot and cold inlet flowmeters was carried out using the following method applied to each flowmeter separately:

1. The *process reaction tank* was emptied.

2. The *outlet valve* was then closed and the tank was filled with a constant inlet flow measured by the PerFlow visual flow meter. During this time the flow signal from the flow meter to be calibrated, the main process tank level signal and the time were logged.

3. When the *level* in the tank reached 150cm the measurement was terminated.

4. This *procedure* was carried out separately for each flow inlet at selected flow values over the full range of inlet flows.

Using the data gathered in the above test the volume of water that flowed into the tank during each trial can be calculated in two ways :

- calculation of the change in volume using the start and finish levels

- integration of the flow signals over the time period of the measurement

These two methods are represented by (4.1) and (4.2).

$$V_{Level} = \pi r^2 \, (Level_{Finish} - Level_{Start}) \qquad (4.1)$$

$$V_{Flow} = \int_{0}^{tfinish} Flow(t)\, dt \qquad (4.2)$$

where V_{Level} is the tank volume calculated by the level method,
V_{Flow} is the tank volume calculated by the flow method,
r is the tank radius,
$Level_{Finish}$ is the final tank level value,
$Level_{Start}$ is the initial tank level value,
$Flow(t)$ is the flow of flowmeter to be calibrated and
tstart,tfinish are the times required for the experiment.

For a correctly calibrated flow sensor, the two volumes V_{Level} and V_{Flow}, should be equal, assuming a constant flow rate between data samples. Application of (4.3) to the data from the flow sensors allows a correct recalibration:

$$F_{cal}(t) = \frac{V_{Level}}{V_{Flow}} F_{data}(t) \qquad (4.3)$$

where $F_{cal}(t)$ is the calibrated flow value and
$F_{data}(t)$ is the uncalibrated flow data.

The data collected in this experiment is listed in Table 1 in Appendix C.

To interpolate between the values delivered by the ADC and the physical flow values using the units m^3s^{-1}, polynomials were fitted to the data derived from the re calibration. The *polynomials* were obtained from the MATLAB 4.0 [98] function *polyfit()* - equations (4.4) and (4.5). The order of polynomial (4.4) was set to first order - this was based on the Order of the Measurement Principle. The polynomial used for the determination of the cold inlet flow is third order. This contrasts with the order of the physical measurement principle - but gives superior accuracy. These polynomials replaced the initial characteristics and were used for all further experiments in this research. Figure 4.1 (see page 63) depicts the characteristics for each flowmeter obtained from the empirical method described above.

$$F_{hot}(\mathbf{X}) = 0.35866\ \mathbf{X} \qquad (4.4)$$

where $\mathbf{X}$ is the value from the ADC and
$F_{hot}(\mathbf{X})$ is the calculated flow for the hot inlet.

$$F_{cold}(\mathbf{X}) = 1.3307\text{x}10^{-10}\ \mathbf{X}^3 - 3.5008\text{x}10^{-7}\ \mathbf{X}^2 + 3.9994\text{x}10^{-4}\ \mathbf{X} \qquad (4.5)$$

where $F_{cold}(\mathbf{X})$ is the calculated flow for the cold inlet.

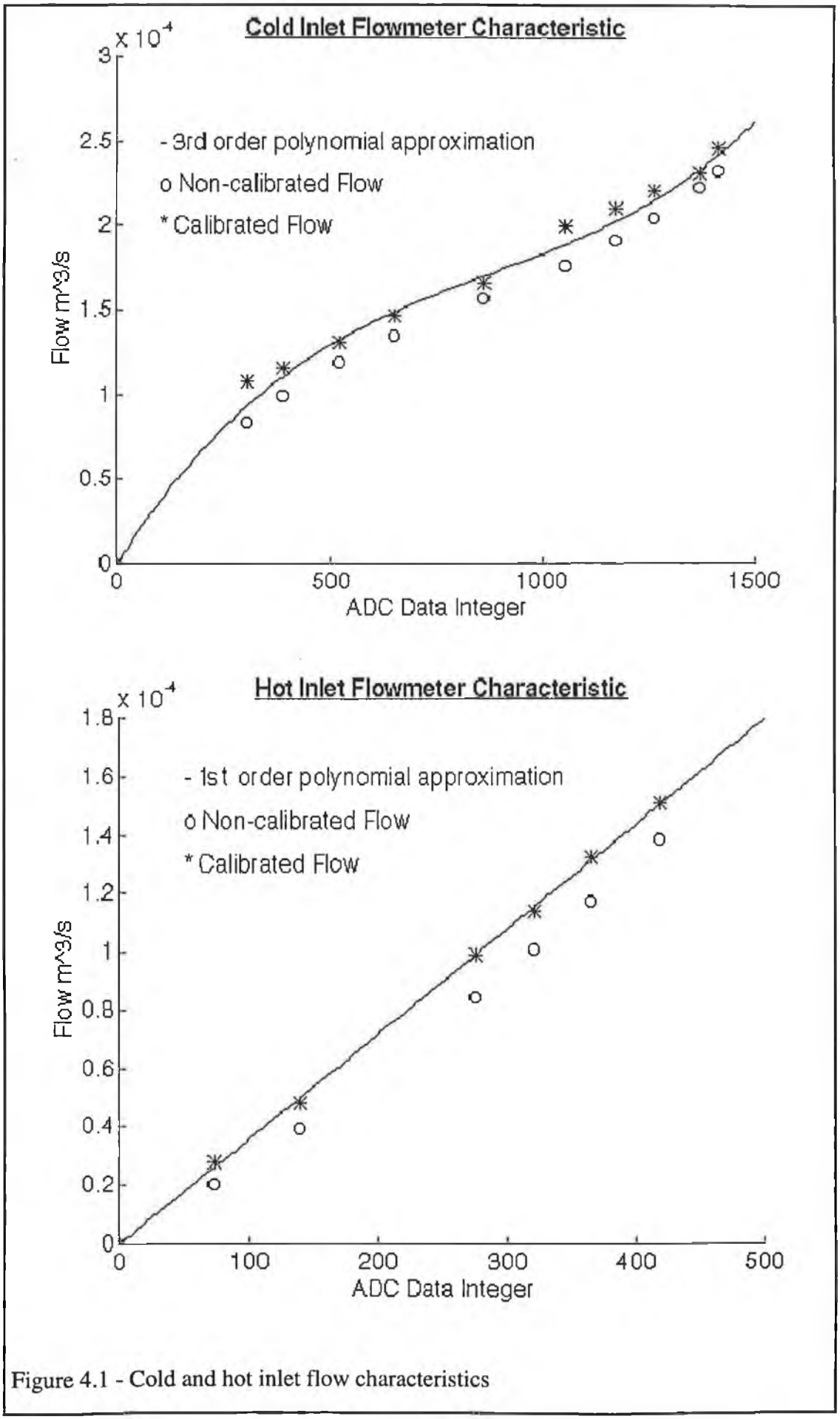


Figure 4.1 - Cold and hot inlet flow characteristics

In order to calibrate the outlet flow meter, the data for the valve position of 100% taken from the experiment described in Section 4.5.1.1 was used. Both (4.1) and (4.2) were applied to this data set. Based on the physical measurement a linear characteristic for the outlet flow meter was assumed. The calibrated outlet flowmeter characteristic is given by (4.6) and was derived from the results given by (4.1) and (4.2).

$$F_{out}(\mathbf{X}) = 0.972\ \mathbf{X} \tag{4.6}$$

where *Fout*(**X**) is the outlet flow value.

4.3. Valve Linearisation

As described in Chapter 3, three valves are used to regulate the inlet and outlet flows of the plant. In order to be able to maintain inlet flow at constant values a closed loop control strategy was decided upon. Proportional-Integral-Derivative (PID) controllers were thus designed for the hot and cold inlets. As the time constants of the inlet valves are of the order of 0.5 seconds, it was necessary to execute these controllers within the interrupt service routine, as described in Chapter 3, at a sampling time of 0.05s.

The idealised continuous time form of the PID controller is given by (4.7).

$$u(t) = K\left[e(t) + \frac{1}{T_I}\int_0^t e(\tau)d\tau + T_D\frac{de(t)}{dt}\right] \tag{4.7}$$

where $u(t)$ is the controller output value,
$e(t)$ is the error value (controller input),
K is the proportional gain,
T_I is the integral time constant and
T_D is the derivative time constant.

Based on the equations for digital PID algorithms given by Åstrom and Wittenmark [99] digital PID controllers were implemented within the interrupt service routine. The proportional, integral and derivative terms are all calculated separately and the controller output is the sum of all three. Equation (4.8) shows the calculation used for the proportional term $u_P(n)$.

$$u_P(n) = K\left[bU_c(n) - y(n)\right] \tag{4.8}$$

where $U_c(n)$ is the setpoint,
$y(n)$ is the output from the plant and
b is a gain less than or equal to unity.

The integral term $U_I(n)$ is given by the equation (4.9).

$$u_I(n) = u_I(n-1) + \frac{KT_S}{T_I} e(n-1) \tag{4.9}$$

where Ts is the sampling time and
$e(n-1)$ is the previous error value.

The derivative term of the PID controller is given by equation (4.10). The gain of the derivative term is limited at high frequencies and the plant output is differentiated instead of the plant error - thus avoiding disturbances from setpoint changes.

$$u_D(n) = \frac{T_D}{T_D + NT_S} u_D(n-1) - \frac{KT_D N}{T_D + NT_S}\left[y(n) - y(n-1)\right] \tag{4.10}$$

where N is a gain of between 3 and 20.

$$u(n) = u_P(n) + u_I(n) + u_D(n) \tag{4.11}$$

The output of the controller, $u(n)$, is given by (4.11). A problem often encountered with the PID controller is *integral windup*. One simple solution for integral windup is to stop updating the integral term of the PID controller if the value of *controller output* exceeds predefined boundaries. These limiting boundaries can be set to correspond to actuator saturation. This solution to integrator windup was implemented along with the PID algorithm as given by (4.8,9,10,11) for the control of the hot and cold inlet flows.

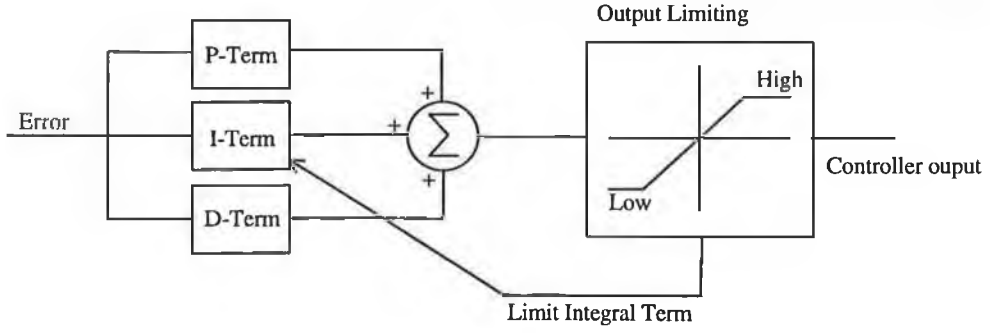


Figure 4.2 - PID controller structure for inlet valve flow control.

To facilitate the design of these PID controllers it was necessary to investigate the steady state response of each of the two inlet valves to current signals of various amplitudes. Thus the following experiment was carried out for each of the inlet valves:

1. The *main tank* was emptied.

2. The *signal* to the valve of the inlet under test was set to a value of zero.

3. Every 30 seconds the *valve control signal* was incremented by the integer of value of 50 (over the DAC output range of 0 to 4095) and the inlet flow was measured.

4. For each *increment* the value of flow through the inlet was logged. Incrementation ceased after the DAC output value of 4095 was reached (maximum value).

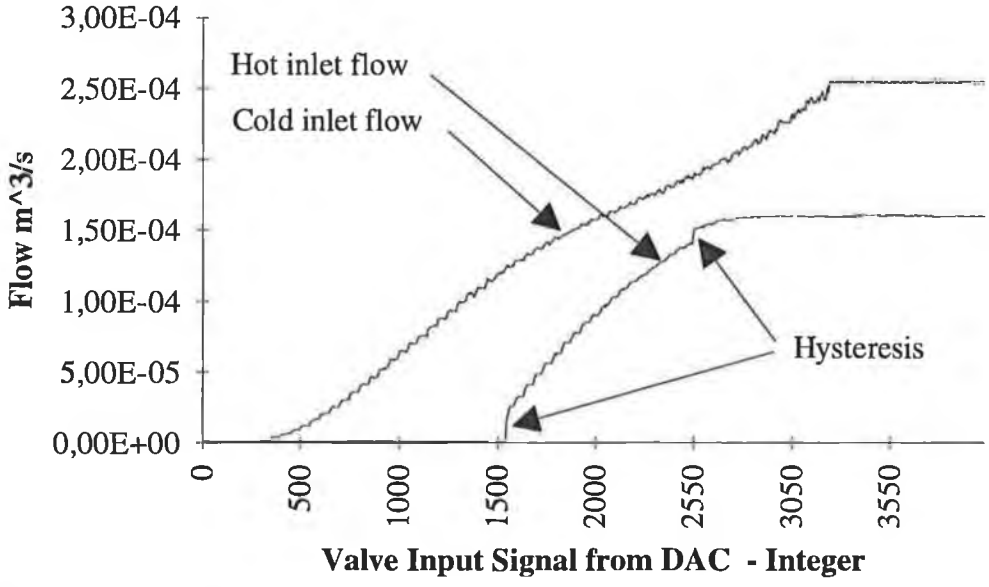


Figure 4.3 - Steady state flow characteristics for the hot and cold inlet flows.

Figure 4.3 shows the steady state characteristics of the hot and cold inlets. Based on the results of these experiments the limit values of the PID controller outputs for the anti-integral windup mechanism for the inlet valves were determined. These values are listed in Table 4.1. Figure 4.2 (see page 65) shows the structure of the PID controller used for control of the hot and cold inlet flows.

The controller constants were adjusted empirically on the actual warm water process by first setting the integral and derivative gains to zero, slowly increasing the proportional gain and observing the corresponding inlet step response. The integral gains were adjusted in turn. As the responses of these PI controllers were adequate, the derivative gains were not used. Figures 4.4, 4.5 and 4.6 show the responses of both the hot and cold inlet PI controllers to setpoints of 20 ml/s, 100 ml/s and 140 ml/s. The values of the controller parameters thus achieved are listed in Table 4.2 (see page 65). The values of the controller parameters of b and N as given in (4.8) and (4.10) were 1 and 4 respectively.

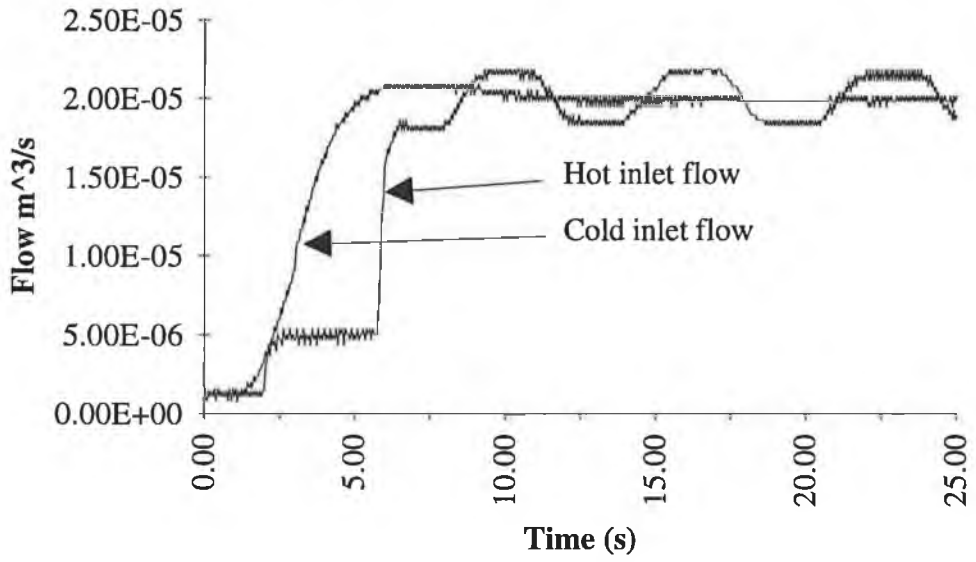


Figure 4.4 - Hot and cold inlet PI controller responses for a flow setpoint of 20 ml/s from the warm water process.

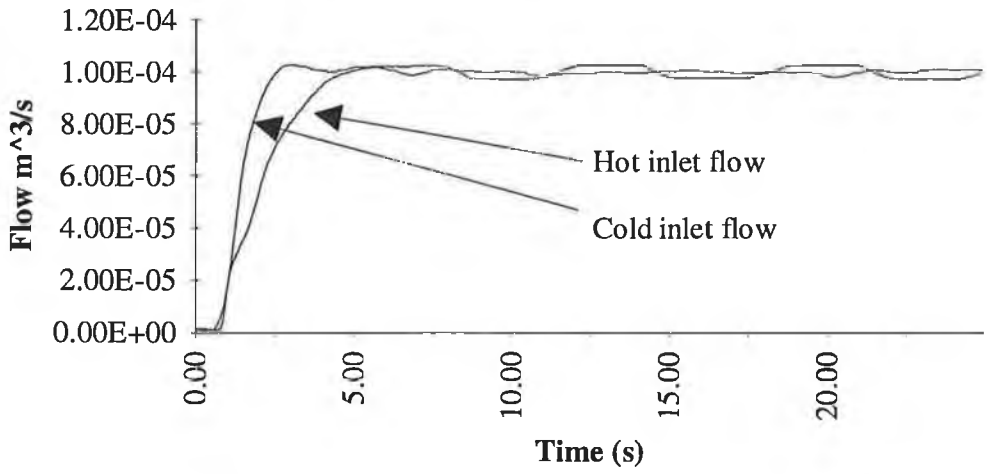


Figure 4.5 - Hot and cold inlet PI controller responses for a flow setpoint of 100 ml/s from the warm water process.

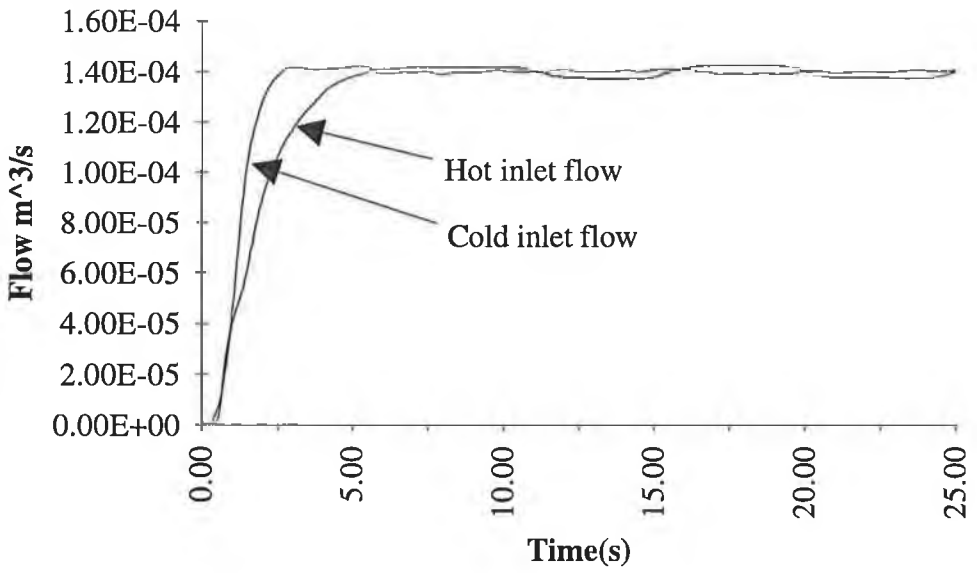


Figure 4.6 - Hot and cold inlet PI controller responses for a flow setpoint of 140 ml/s from the warm water process.

Table 4.1 - Limit values for integral windup - DAC integer values.

	Lower Limit U_{low}	Upper Limit U_{high}
Hot Inlet	150	3000
Cold Inlet	50	3500

Table 4.2 - PID controller parameters for the hot and cold inlets.

	Proportional Gain	Integral Gain	Derivative Gain
Hot inlet	4×10^6	3.0769×10^5	0
Cold inlet	5×10^6	1.25×10^5	0

The response of each of the inlet PI controllers differs around various operating points. This is due to the non-linearity of the inlets arising from :

- *Deadtime* - each inlet has an inherent deadtime due to the physical distance between the valve and the flowmeter. This deadtime is inversely proportional to the flowrate as can be seen by close examination of the controller responses in Figures 4.4 - 6. The deadtime for a flowrate of 20 ml/s, assuming a distance of 20cm between the valve and flowmeter, is approximately 1.5 seconds.

- *Non-Linear Valve and Flowmeter Characteristics* - both inlet valves and flowmeters have non-linear steady state characteristics as can be seen clearly in Figure 4.3 (see page 66).

- *Hysteresis* - this non-linearity is typically found in mechanical systems and thus contributes to the non-linearity of the inlet valves. Hysteresis of the hot inlet valve and turbine flow meter can easily be seen in Figure 4.3 (see page 66).

4.4. Choice of the Sampling Time for the Warm Water Process

In order to collect data from the warm water process, a data acquisition program was written. For this data acquisition programme, the interrupt driven software structure as described in Section 3.5 combined with the PI controllers from Section 4.3 were utilised. An important parameter for the data acquisition and for warm water process control, is the sampling rate. In order to determine the suitable sampling rate and to gain more insight into the dynamic behaviour of the warm water process, an initial set of step tests on the warm water process were carried out. Initially a sampling rate of two seconds was used for these tests. This value was chosen as its maintains a good signal to noise ratio and, based on the physical model of the warm water process, is small enough to allow the observation of the plant dynamics.

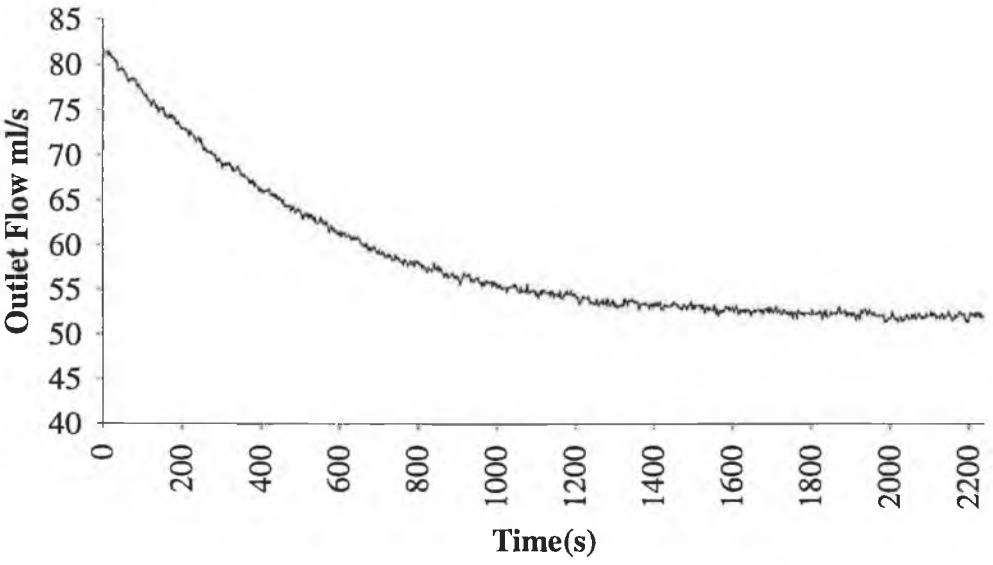


Figure 4.7 - Response of the outlet flow of the process reaction tank to a step input of 40 ml/s at the hot inlet flow.

Figures 4.7 and 4.8 show the outlet flow and temperature responses for a step test of 40 ml/s at the hot inlet. Assuming first order with dead time system responses, the time constants for outlet flow and temperature from these responses are 540 and 594 seconds respectively. Thus a sampling time of 30 seconds was decided upon for data acquisition from and closed loop control of the warm water process. This value is

smaller than one tenth of the dominant time constants of both the outlet flow and temperature variables.

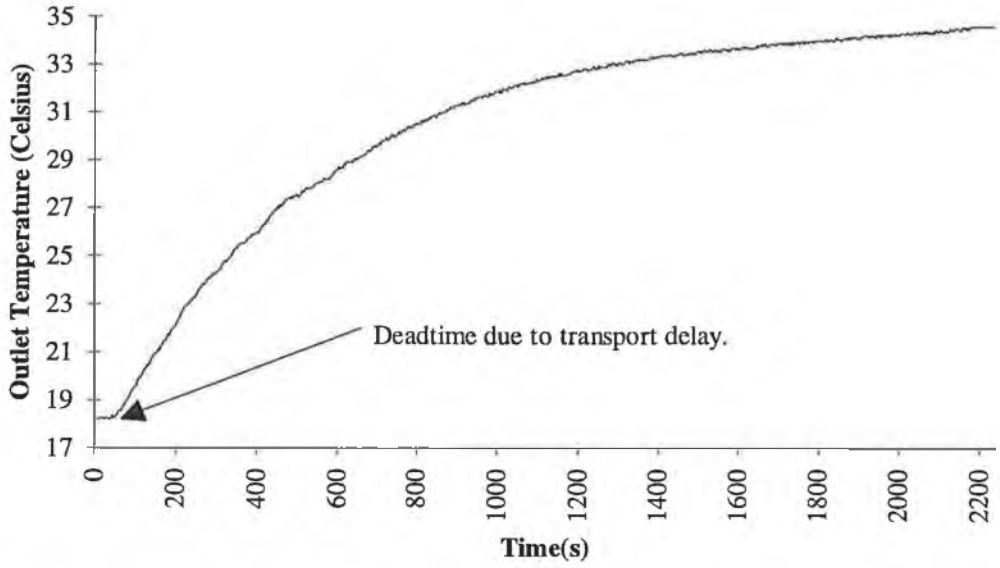


Figure 4.8 - Response of the outlet temperature of the process reaction tank to a step input of 40 ml/s at the hot inlet flow.

4.5. Warm Water Process Model from Physical First Principles

The warm water process can be described by a set of mass flow and thermal energy equations which can be derived from physical first principles. These physical first principles model equations have a given structure whereby the equation parameters must be either mathematically derived or determined through suitable tests on the warm water process. The warm water process can be described as multivariable with coupling between the outlet variables i.e. outlet flow and outlet temperature. The general form of the physical first principles model is given by (4.12)

$$\begin{aligned} \dot{F}_{out}(t) &= f_1\{F_{cold}(t),\ F_{hot}(t)\} \\ \dot{T}_{out}(t) &= f_2\{F_{cold}(t),\ F_{hot}(t)\} \end{aligned} \tag{4.12}$$

where $\dot{F}_{out}(t), \dot{T}_{out}(t)$ are the first derivatives of the outlet variables,
f_1, f_2 are the functions of the inlet flows

This section describes firstly the derivation of the physical first principle modelling equations for both the mass flow and the thermal energy behaviour of the warm

process. Following this, the experiments used to determine some of the modelling parameters are iterated. Finally, a description of the programmed simulation of the warm water process in the MATLAB/SIMULINK environment and comparisons with real plant behaviour are presented.

4.5.1. Mass Flow Equations

The mass flow behaviour of the process reaction tank and its outlet section including pipe line, flow transducer and outlet valve under the assumption of lumped outlet losses and turbulent flow can be described by (4.13) and (4.14), see reference [100].

$$F_{out}(t) = C_v A_2 \sqrt{2g\,h(t)} \tag{4.13}$$

$$A_{tank} \frac{dh(t)}{dt} = F_{cold}(t) + F_{hot}(t) - F_{out}(t) \tag{4.14}$$

where g is the acceleration due to gravity,
C_v is the coefficient of discharge of the process reaction tank,
A_2 is the cross sectional area of outlet of the process reaction tank,
A_{tank} is the cross sectional area of the process reaction tank and
$h(t)$ is the head of water in the process reaction tank.

The constant C_v is the coefficient of discharge of the process reaction tank and is dependent on the losses in the outlet line including the outlet valve, outlet flowmeter and the outlet pipe when these are lumped at the process reaction tank outlet. The parameter A_2 represents the effective area of the tank outlet. Due to the lumped losses assumption, this value is not only a function of the tank outlet diameter but also of the outlet valve position. The parameter A_{tank} is the area of the tank in square metres and was calculated from the measured diameter of the tank.

4.5.1.1. Determination of Mass Flow Parameters C_v and A_2

To investigate the validity of the mass flow equations as detailed in Section 4.5.1 and to calculate the value of the product of the mass flow parameters C_v and A_2 the following experiment was carried out on the warm water process for four different outlet valve positions:

1. The *outlet valve* of the process reaction tank was closed.

2. The *process reaction tank* was filled to a level of approximately 150cm.

3. The position of the process reaction tank *outlet valve* was set to 100%.

4. The *process reaction tank* was allowed to *empty* while the level *h(t)* and outlet flow $F_{out}(t)$ data were logged.

5. Steps 1 through 4 were repeated for the other three process reaction tank outlet valve positions of 75%, 50% and 25%.

From the results obtained it was possible to calculate the C_vA_2 value using equation (4.15) which is derived from (4.13).

$$C_vA_2 = \frac{F_{out}(t)}{\sqrt{2g\,h(t)}} \tag{4.15}$$

Figures 4.9, 4.10, 4.11 and 4.12 show the experimental values of C_vA_2 for the four process reaction outlet valve positions obtained through the application of (4.15) to the data gathered in this experiment. The large fluctuations at low flow and head values are due to the decrease in the signal to noise ratio of the measurements at lower levels. The saw tooth appearance of the characteristics is attributable to the 2cm resolution of the level meter as described in Section 3.2.3.5. To calculate the average value of the C_vA_2 parameter for each valve position, only values corresponding to tank levels above 50cm were utilised - thus omitting data with poor signal to noise ratio. The average values of C_vA_2 and their standard deviations for all measurements above tank levels of 50cm are listed in Table 4.3.

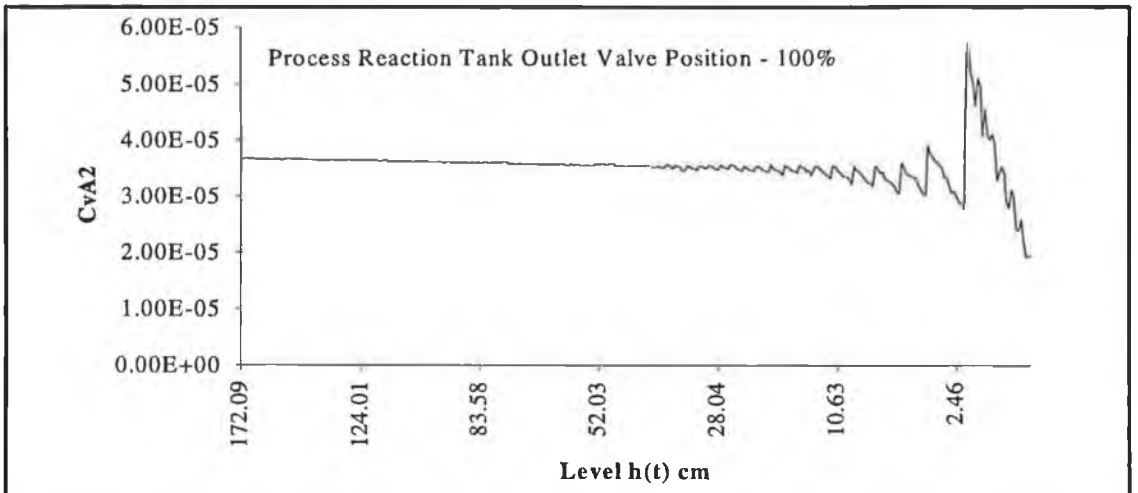


Figure 4.9 - Plot of experimental C_vA_2 values at an outlet valve position of 100%.

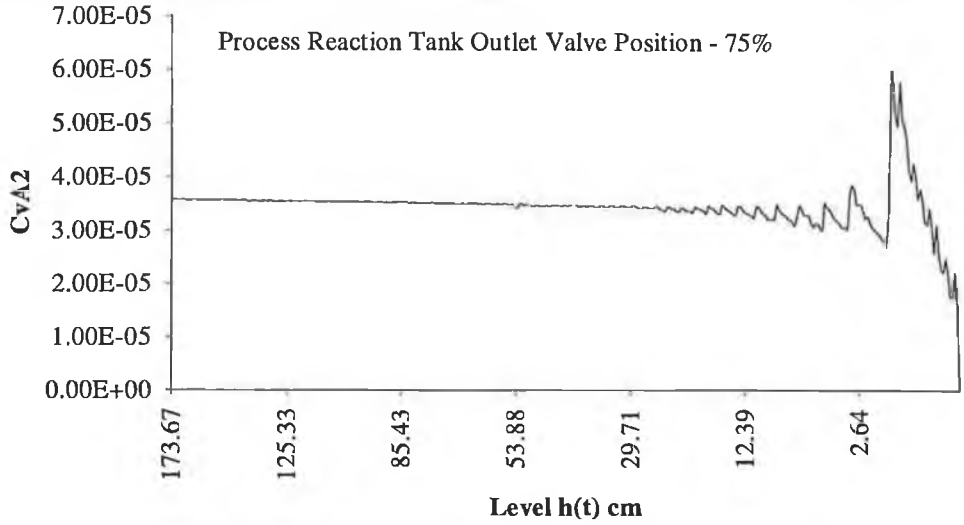


Figure 4.10 - Plot of experimental C_vA_2 values at an outlet valve position of 75%.

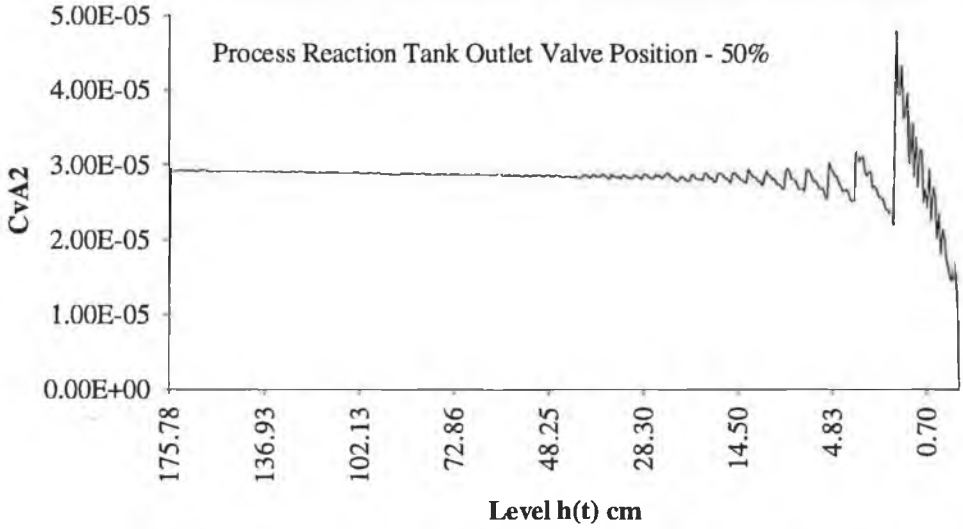


Figure 4.11 - Plot of experimental C_vA_2 values at an outlet valve position of 50%.

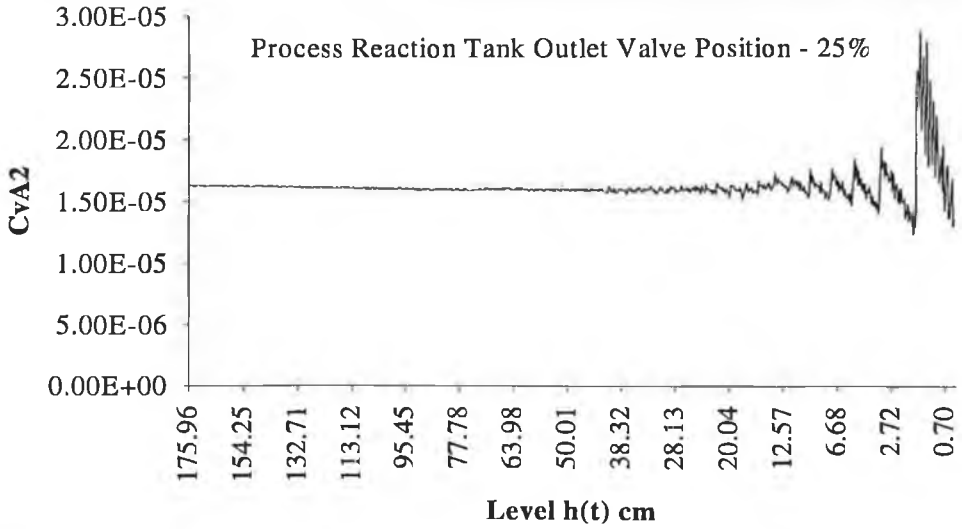


Figure 4.12 - Plot of experimental C_vA_2 values at an outlet valve position of 25%.

Table 4.3 - Table of average C_vA_2 values and their variances.

Process Reaction Tank Outlet Valve Position				
	100%	**75%**	**50%**	**25%**
Mean	3.62E-05	3.53E-05	2.88E-05	1.61E-05
Variance	3.58E-07	3.15E-07	2.19E-07	1.41E-07

Using the average values obtained for C_vA_2 from the four experiments described above, a second order polynomial function was fitted to the four values using the MATLAB *polyfit()* function. A second order polynomial was chosen as it corresponds to the order of (4.15), which represents the physical relationship between the outlet flow $F_{out}(t)$ and the value of C_vA_2, assuming a linear outlet valve characteristic. This polynomial is given by equation (4.16) where the argument $\mathbf{V}_{pos}$ is the process reaction tank outlet valve position in percent.

$$C_vA_2(\mathbf{V}_{pos}) = -4.1739\times10^{-9}\ \mathbf{V}_{pos}^{\ 2}\ \ 7.7567\times10^{-7}\ \mathbf{V}_{pos}\ -2.0275\times10^{-7} \qquad (4.16)$$

The polynomial (4.16) thus allows the calculation of C_vA_2 given the position of the process reaction tank outlet valve in percent. Figure 4.13 shows the characteristic obtained from (4.16) together with the four experimental values of C_vA_2.

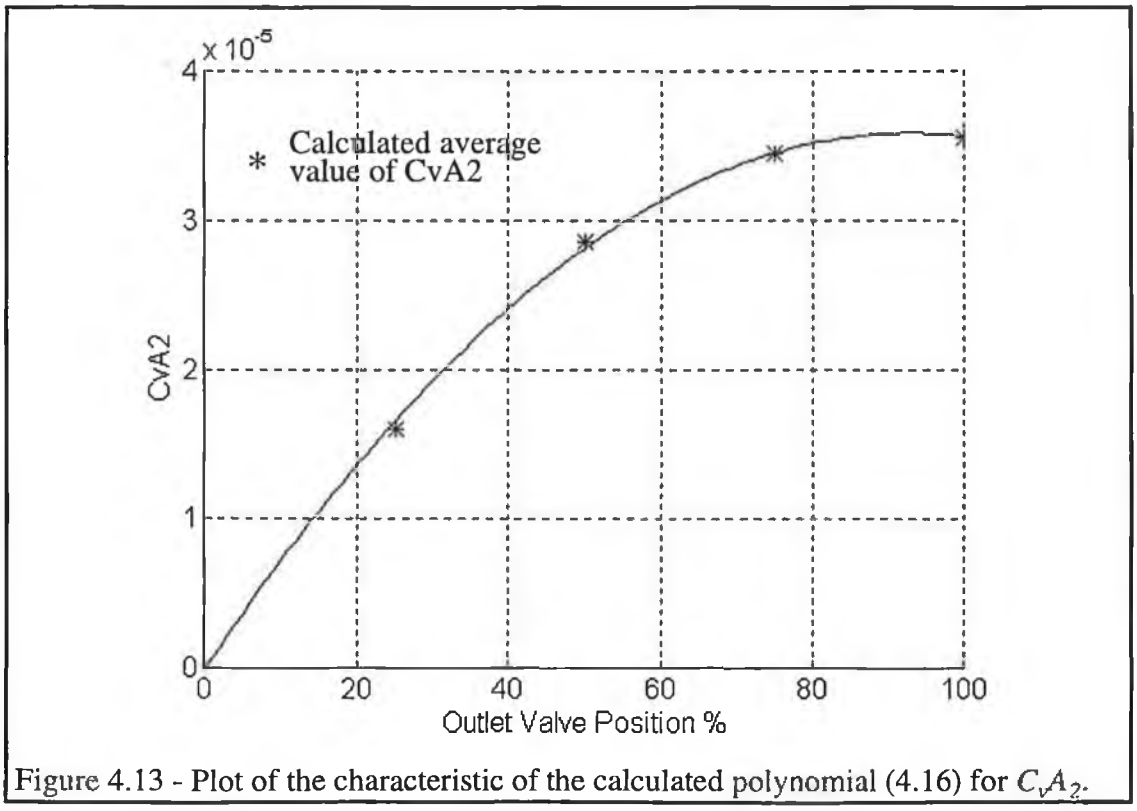


Figure 4.13 - Plot of the characteristic of the calculated polynomial (4.16) for C_vA_2.

4.5.2. Thermal Energy Equations

The thermal equations for the plant are non-linear and dominated by product terms of the inlet flow and temperature variables. Assuming perfect mixing and no heat losses within the process reaction tank, the following details the derivation of the physical first principle thermal equations.

$$\frac{d\,F_{out}(t)}{dt} = F_{hot}(t) + F_{cold}(t) - F_{out}(t) \tag{4.17}$$

Equation (4.17) represents the mass flow of the process reaction tank. The change in heat within the process reaction tank due to an input flow $F_{in}(t)$ is given by (4.18).

$$\therefore \frac{d\,Q_{tank}(t)}{dt} = \rho\, c\, F_{in}(t)\ \frac{d\,\theta(t)}{dt}$$

$$= \rho_{hot}\, c\, F_{hot}(t)\, \theta_{hot}(t) + \rho_{cold}\, c\, F_{cold}(t)\, \theta_{cold}(t) \tag{4.18}$$

where ρ is the density of inflowing water,
ρ_{hot} is the density of hot water,
ρ_{cold} is the density of cold water,
c is the specific heat capacity of water,
$\theta_{hot}(t)$ is the temperature of hot water,
$\theta_{cold}(t)$ is the temperature of cold water and
$Q_{tank}(t)$ is the heat within the process reaction tank.

Assuming constant density ρ for the hot and cold inlet flows and constant head $h(t)$, with $V_s = h(t)\, A_{tank}$ where V_s is the process reaction tank volume, gives :

$$V_s\, \frac{d\theta_{tank}(t)}{dt} = F_{hot}(t)\, \theta_{hot}(t) + F_{cold}(t)\, \theta_{cold}(t) \tag{4.19}$$

Equation (4.19) can be augmented by considering both heat loss and non-perfect mixing within the process reaction tank. Heat loss through the walls of the process reaction tank is described by a non-linear function of the temperature difference between the temperature of the water in the tank and the ambient temperature and the level of liquid in the tank. Mixing dynamics are also described by a non-linear function of all flow and temperature variables as well the level of fluid in the tank. Thus the complete equation for heat flow in the system from first principles is given by (4.20).

$$V_s \frac{d\theta_{out}(t)}{dt} = F_{hot}(t)\theta_{hot}(t) + F_{cold}(t)\theta_{cold}(t) - Q_{loss}(t) + Q_{mix}(t) \quad (4.20)$$

where $\theta_{out}(t)$ is the process reaction tank outlet temperature,

$$Q_{loss}(t) = f(\theta_{tank}(t),\ \theta_{ambient}(t),\ h(t)) \text{ and} \quad (4.21)$$

$$Q_{mix}(t) = g\left(Q_{hot}(t),\ Q_{cold}(t),\ Q_{tank}(t),\ h(t),\ F_{hot}(t),\ F_{cold}(t),\ F_{out}(t)\right) \quad (4.22)$$

The physical model used in this research assumes perfect mixing. The accuracy of this set of equations based on physical first principles is to be investigated through suitable experiments on the plant. This investigation is described in Section 4.5.4.

4.5.3. MATLAB/SIMULINK Simulation of the Physical Warm Water Process Model

The physical equations of the warm water plant as detailed earlier in this chapter were simulated within the MATLAB/SIMULINK environment. Due to the process specific non-linear nature of the equations, user defined *s-functions* were written in the MATLAB/SIMULINK macro programming language and then incorporated into a MATLAB/SIMULINK icon. The models use the empirically derived parameters detailed in this chapter. The macro code used for this simulation icon is contained in Appendix D.

4.5.4. Validation of the Physical Model

To determine the nature of the accuracy of the simulated warm water process physical model, a *series of twelve step tests for each of the outlet valve positions of 25, 50, 75 and 100%* were performed on the warm water process. Before running these tests the hot inlet and the process reaction tank outlet valves were fully opened until the temperature in the hot reservoir tank reached an initial temperature of 40°Celsius. This ensured that hot water was available for the step tests within the process reaction tank, whereby the temperature of the water in the hot reservoir tank is fully dependent on the temperature of the hot water supply. The choice of the temperature of 40°C was based on experience with the warm water process, where the hot water supply temperature rarely rises above 45°C. The tests performed for each process reaction outlet valve position are listed in Table 4.4 (see page 81). The data from each of these tests was used for two purposes :

- to *serve as test data* for evaluation of the artificial neural network model developed in Section 4.6,

- to *evaluate the accuracy* of the first principles model of the process.

The data from the step tests for a process reaction tank outlet valve position of 100% were used to evaluate the accuracy of the simulated physical model of the warm water process in the following manner :

1. The *step test* data was loaded into the MATLAB workspace.

2. The inlet flow and temperature data, i.e. $F_{cold}(t)$, $F_{hot}(t)$ and $T_{hot}(t)$, were used to drive the MATLAB/SIMULINK model of the warm water process and the output data, i.e. $F_{out}(t)$, $h(t)$ and $T_{out}(t)$, of the simulation model was logged to the MATLAB workspace.

3. The *output data* from the simulation was then compared with the experimental output data from the step tests.

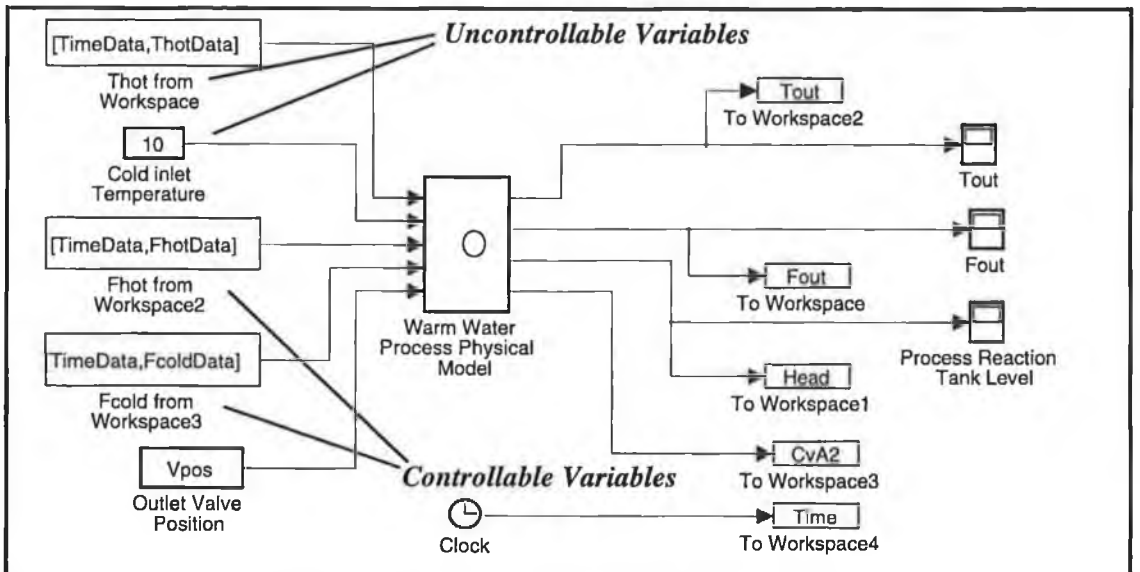


Figure 4.14 - MATLAB validation of the warm water process physical model.

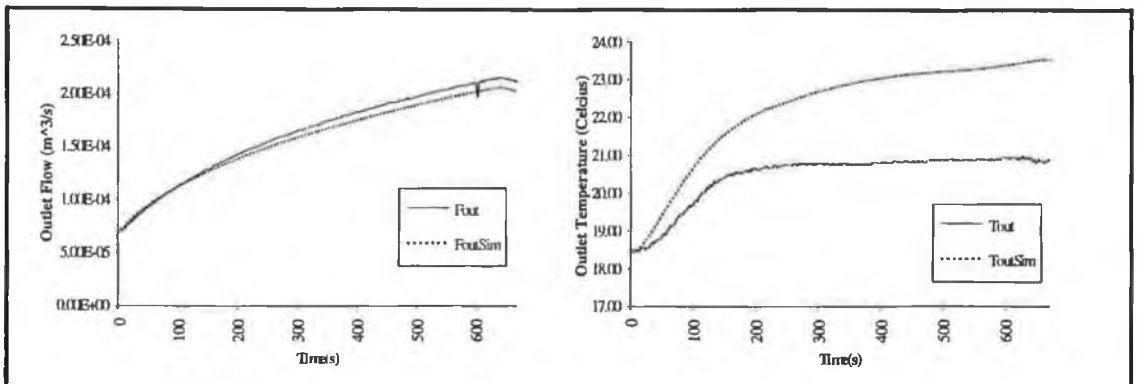


Figure 4.15 - Step Test 37 : $F_{cold}(t)$ = 240 ml/s and $F_{hot}(t)$ = 140 ml/s.

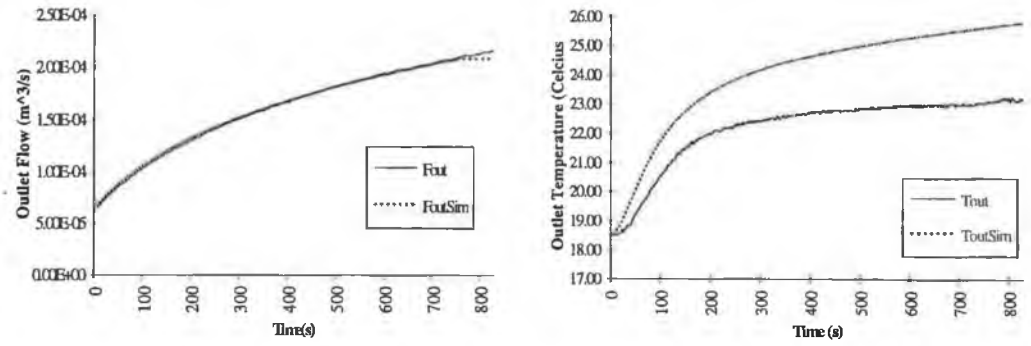


Figure 4.16 - Step Test 38 : $F_{cold}(t)$ = 160 ml/s and $F_{hot}(t)$ = 140 ml/s.

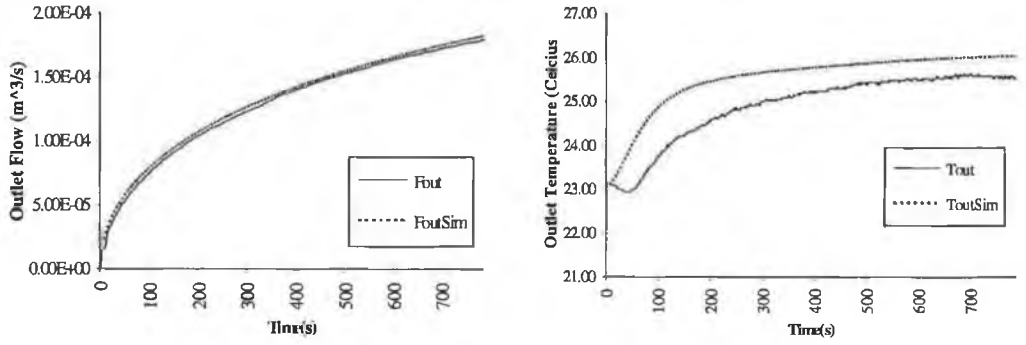


Figure 4.17 - Step Test 39 : $F_{cold}(t)$ = 80 ml/s and $F_{hot}(t)$ = 140 ml/s.

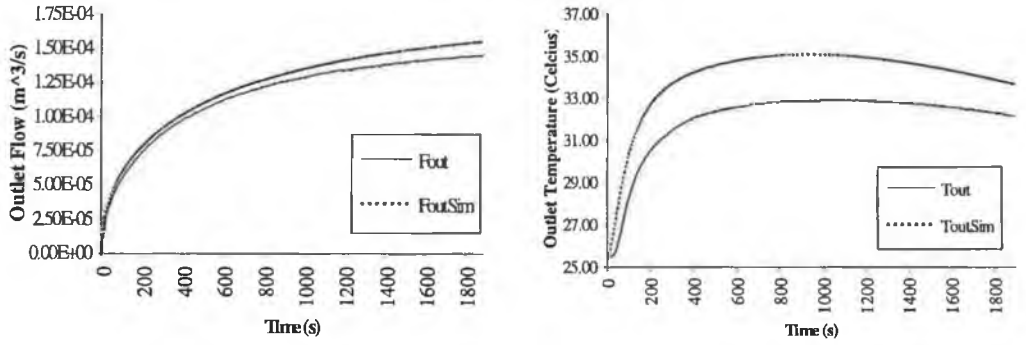


Figure 4.18 - Step Test 40 : $F_{cold}(t)$ = 0 ml/s and $F_{hot}(t)$ = 140 ml/s.

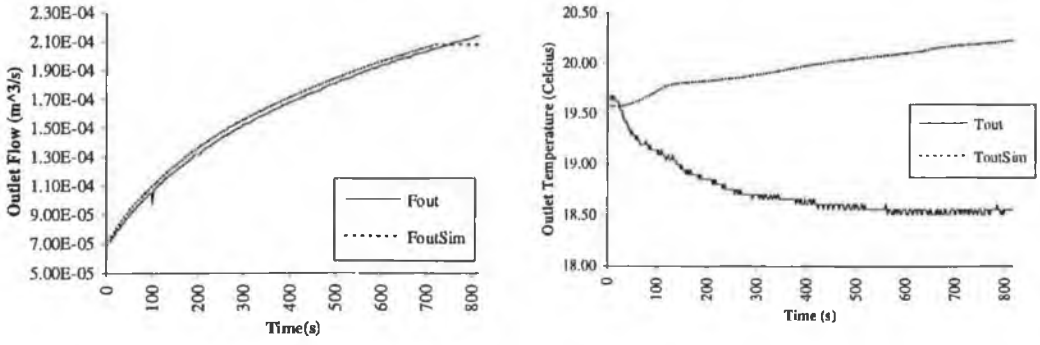


Figure 4.19 - Step Test 41 : $F_{cold}(t)$ = 240 ml/s and $F_{hot}(t)$ = 90 ml/s.

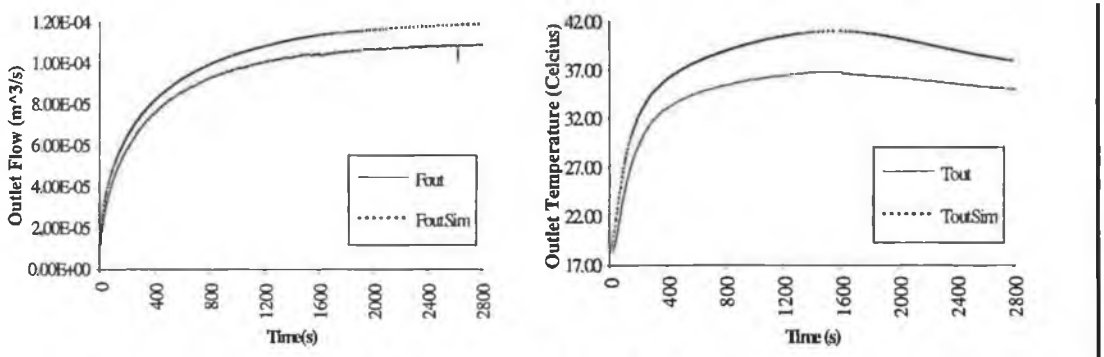


Figure 4.22 - Step Test 44 : $F_{cold}(t)$ = 0 ml/s and $F_{hot}(t)$ = 90 ml/s.

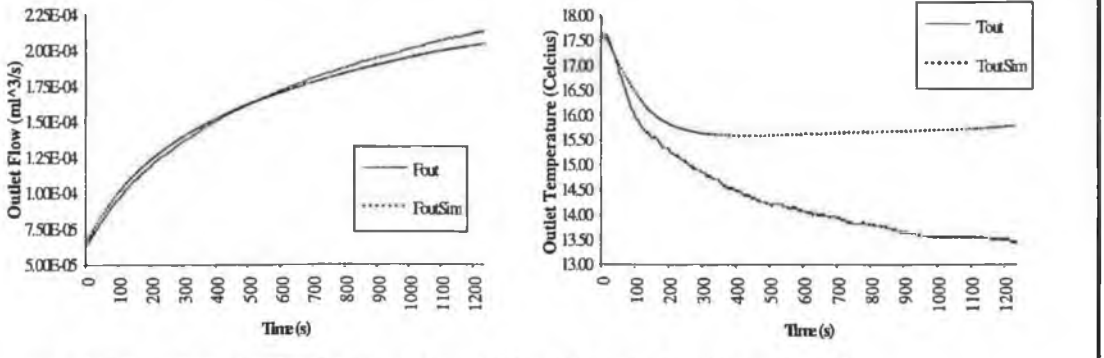


Figure 4.23 - Step Test 45 : $F_{cold}(t)$ = 240 ml/s and $F_{hot}(t)$ = 40 ml/s.

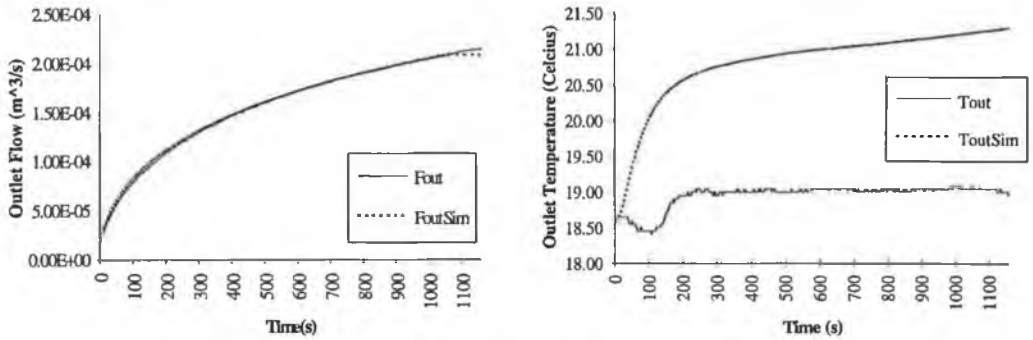


Figure 4.20 - Step Test 42 : $F_{cold}(t)$ = 160 ml/s and $F_{hot}(t)$ = 90 ml/s.

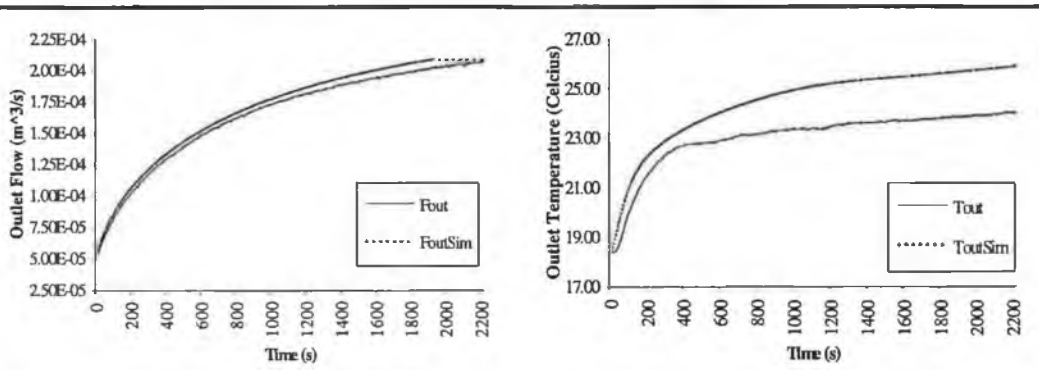


Figure 4.21 - Step Test 43 : $F_{cold}(t)$ = 80 ml/s and $F_{hot}(t)$ = 90 ml/s.

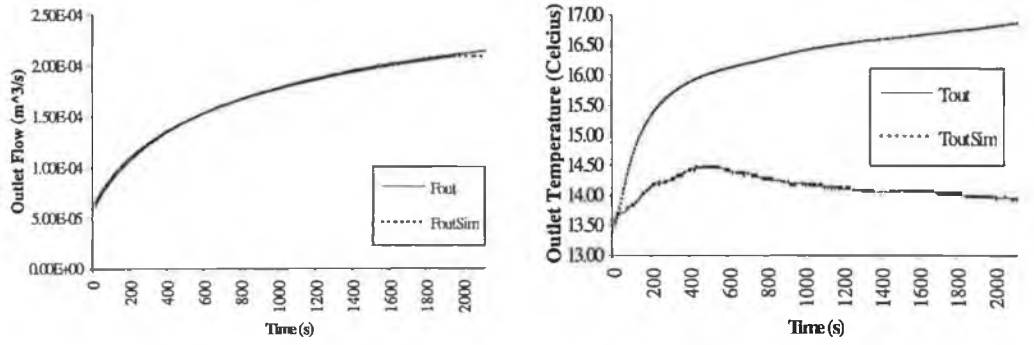


Figure 4.24 - Step Test 46 : $F_{cold}(t)$ = 160 ml/s and $F_{hot}(t)$ = 40 ml/s.

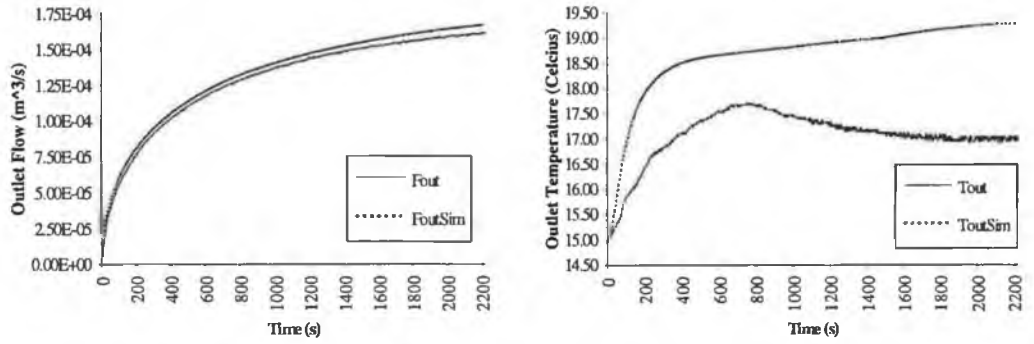


Figure 4.25 - Step Test 47 : $F_{cold}(t)$ = 80 ml/s and $F_{hot}(t)$ = 40 ml/s.

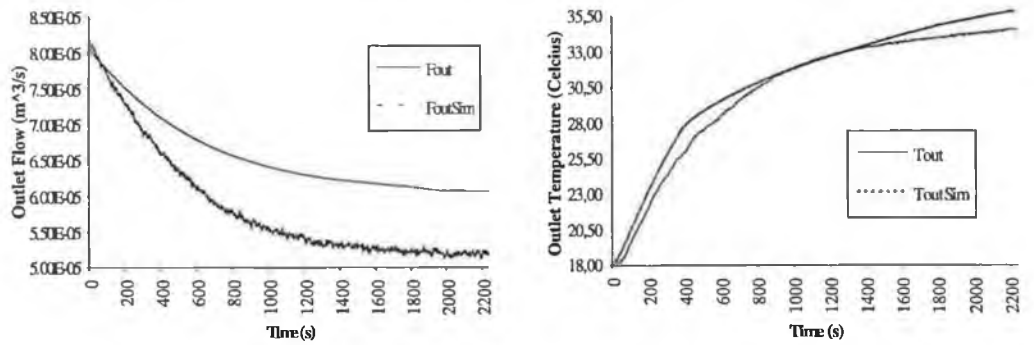


Figure 4.26 - Step Test 48 : $F_{cold}(t)$ = 0 ml/s and $F_{hot}(t)$ = 40 ml/s.

The MATLAB/SIMULINK simulation used for the evaluation of the physical model of the warm water process is shown in Figure 4.14 (see page 77) . The Figures 4.15 - 4.26 contain the characteristics of the warm water process output variables $F_{out}(t)$ and $T_{out}(t)$ for the validation of the physical model. *ToutSim* and *FoutSim* refer to the simulated variables. Subjectively, the mass flow physical model is seen to be relatively accurate for all flows whereas the thermal energy physical model does not offer accurate modelling of the real plant thermal behaviour. An objective criteria for evaluation of the modelling capabilities of the physical models is the root mean square (RMS) modelling errors.

Table 4.4 - Summary of step tests for outlet valve position of 100%.

	Inlet Flow		*RMS Modelling Error*	
NAME	$F_{cold}(t)$ *ml/s*	$F_{hot}(t)$ *ml/s*	$F_{out}(t)$ *ml/s*	$T_{out}(t)$ *Celsius*
Test 37	240	140	6.43	1.91
Test 38	160	140	2.12	1.95
Test 39	80	140	3.12	0.68
Test 40	0	140	6.66	2.01
Test 41	240	90	3.40	1.30
Test 42	160	90	2.11	1.90
Test 43	80	90	4.88	1.50
Test 44	0	90	8.30	3.65
Test 45	240	40	4.49	1.57
Test 46	160	40	2.01	2.18
Test 47	80	40	4.26	1.68
Test 48	0	40	7.77	0.86

The values of the root mean squares of the modelling errors for the physical models are listed in Table 4.4 for all the step tests.

The following conclusions were drawn from this series of tests :

- The *dynamics of the warm water process* are fastest when the output valve is fully open. Thus, in order to keep future experiments within a reasonable time scale, all future experiments were carried out with the outlet valve position at 100%.

- The dynamic and steady state *accuracy of the mass flow model from first principles* was of reasonable quality. The RMS of the modelling error was always less than 8 ml/s which corresponds to 3.33% of the range of the process reaction tank outlet flow.

- The *first principles temperature model* was of poor quality. The RMS of the modelling error had a maximum value of 3.65° Celsius. Moreover, the dynamics of the plant were not even subjectively matched. This was attributed to the assumption of perfect mixing made within the physical model and, to a lesser extent, the non-measurable disturbance variable - the cold inlet temperature.

Although the physical mass flow model gave satisfactory results, the thermal modelling performance of the physical model was unsatisfactory. This inadequate modelling is due to the inability of the thermal physical model to represent the mixing dynamics of the process reaction tank. Due to this inaccuracy of the physical model, two modelling methods from the soft computing field were applied to the warm water process. As both of these methods are capable of modelling non-linear systems through learning the system dynamic behaviour from a set of data, it was hoped that more accuracy would be achieved when modelling the thermal behaviour of the warm water process.

4.6. Artificial Neural Network Model of the Warm Water Process

This section describes the first of the two soft computing modelling methods - the *Artificial Neural Network* (ANN) model. The type of ANN chosen for modelling the warm water process is the *Multi-Layer Perceptron* (MLP). There were two reasons for the choice of this ANN architecture :

- the fact that the *MLP* is *used in more than 85%* of all ANN applications [101],
- It has been shown that a three layer MLP has the *capability* (by adjusting the values of the weights) of modelling any non-linear function to an arbitrary degree of accuracy [102].

The MLP structure is characterised by its feedforward architecture which consists of layers of neurons where a single neuron is connected to each neuron in the following layer by a weight. Unfortunately, few guidelines exist as to the number and type of neurons necessary in each layer to achieve accurate modelling of a function. Moreover, convergence of learning and generalisation properties of the MLP model of the function cannot be guaranteed.

The MLP uses *supervised learning* to learn the function to be modelled from the training data set. Thus in order to train the MLP some form of learning algorithm is necessary. The most commonly used algorithm is the back-propagation algorithm - a modified gradient descent method [103]. As prerequisites for successful modelling using an ANN, the following points should be adhered to :

- *Temporally continuous training data* - the MLP should be trained using temporally continuous data from the system in question.

- *Data from the complete state space* - if the MLP is to model the system over its complete state space, then the training data should be representative of the system state space. Data from around one operating point will only deliver an ANN model suitable for modelling the plant around this one operating point.

Details of various other learning algorithms and network architectures that can used for ANN modelling can be found in [104].

The main hurdle to be overcome before the application of this modelling method to the warm water process was the time required to collect the training data. This is normally not a problem for a plant with short time constants as enough data can be collected over a reasonably short period of time to allow the implementation of a global model. Due to safety restrictions, only fourteen hours of continuous data could be collected from the plant on any one day. This corresponds to 12 setpoint changes and only 1680 data points at a sampling rate of 30 seconds. Using the data acquisition system as described in Section 4.4, data was thus collected over a period of seven days. The desired hot and cold inlet flows were changed randomly every 4000 seconds. This random change was implemented so as to drive the warm water process through as much of its state space as possible in the hope that a global model of the plant could thus be achieved. The process reaction tank outlet valve position was set to 100% for the complete data acquisition process. A total of 9239 training data set vectors were acquired throughout this experiment.

Having acquired the training data, the architecture of the MLP was decided upon. The chosen network architecture was a recursive four layer structure (in order to increase the resolution of the network) of neurons described by the following :

- *Input layer* - as the warm water process is predominantly first order, all variables were represented by their present and previous values. However, due to the non-linear behaviour of the outlet temperature variable, the hot temperature was represented by its present, previous and previous previous values. In addition one past output from each of the variables to be modelled, i.e. outlet temperature and flow, served as inputs thus forming a recursive model. This gave a total of 9 input layer neurons all with unity transfer functions.

- *Two hidden layers* - all neurons in the hidden layers have logarithmic sigmoid function transfer functions. The logarithmic sigmoid function was chosen as all the variables used in the plant are positive. The first hidden layer consisted of 5 neurons whereas the second hidden layer was made up of 10 neurons. The choice

of the number of neurons within the hidden layers was based upon a rule of thumb - first hidden layer approximately one half of the number of input neurons, second hidden layer 2-3 times the number of neurons within the first hidden layer. This rule of thumb has been found to give a good compromise between model accuracy and noise rejection.

- *Output layer* - the final output layer possessed two neurons each with linear transfer functions which correspond to summations of the neuron inputs with the addition of a bias value.

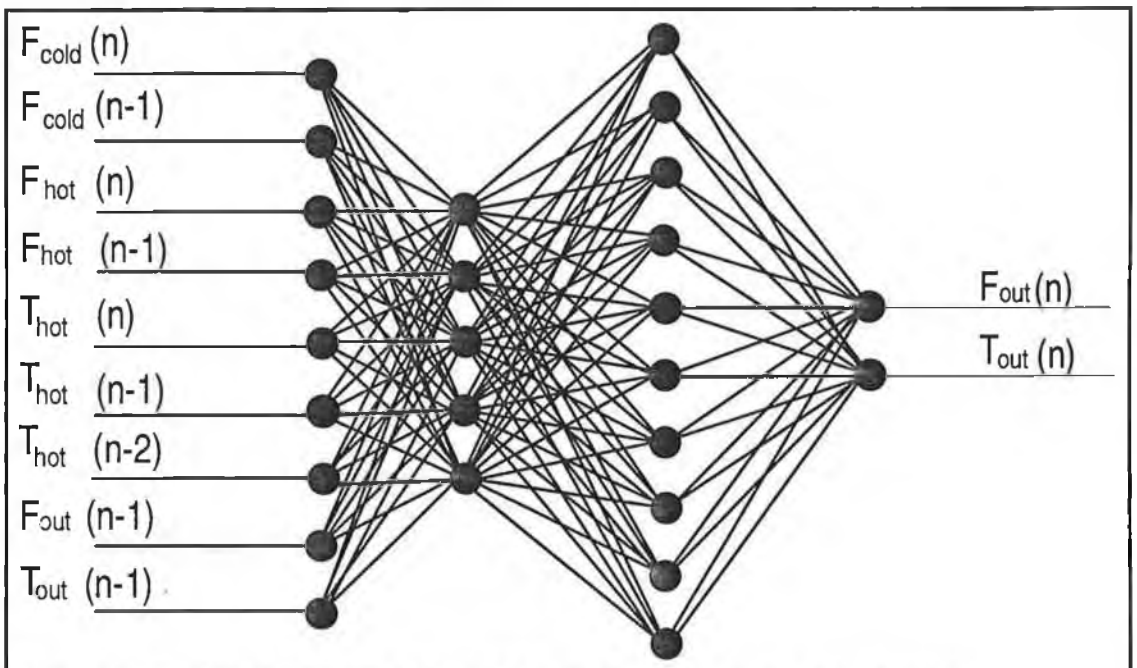


Figure 4.27 - Block diagram of the ANN model of the warm water process.

The MLP model of warm water process is shown in Figure 4.27, with all interconnections between the neurons shown. The chosen learning algorithm for training was backpropagation with a momentum term and an adaptive learning rate [105] which helped to speed learning. With this method the value of the gain used for the update of the weights is adjusted dependent on the rate of decrease of the overall modelling error. For high rates of decrease the learning rate is increased and for is decreased for lower rate of decrease of the modelling error of the ANN. The momentum term is applied to assist the ANN in overcoming local minima which occur due to the non-linear nature of the ANN. After the input and target data were scaled to lie within the [0,1] interval, the network was batch trained for 25,000 epochs. Batch training means that the neuron weights were updated after one complete epoch whereby the sum of the squared error is used for the calculation of the gradients of the weights. The required training time using the MATLAB Neural Net Toolbox

v.1.1. [105] on a 486 DX2 66 MHz PC with 16Mbytes of RAM was three days. Although this training time is long, the comfort and versatility of the MATLAB environment would be lost if faster software tools were used.

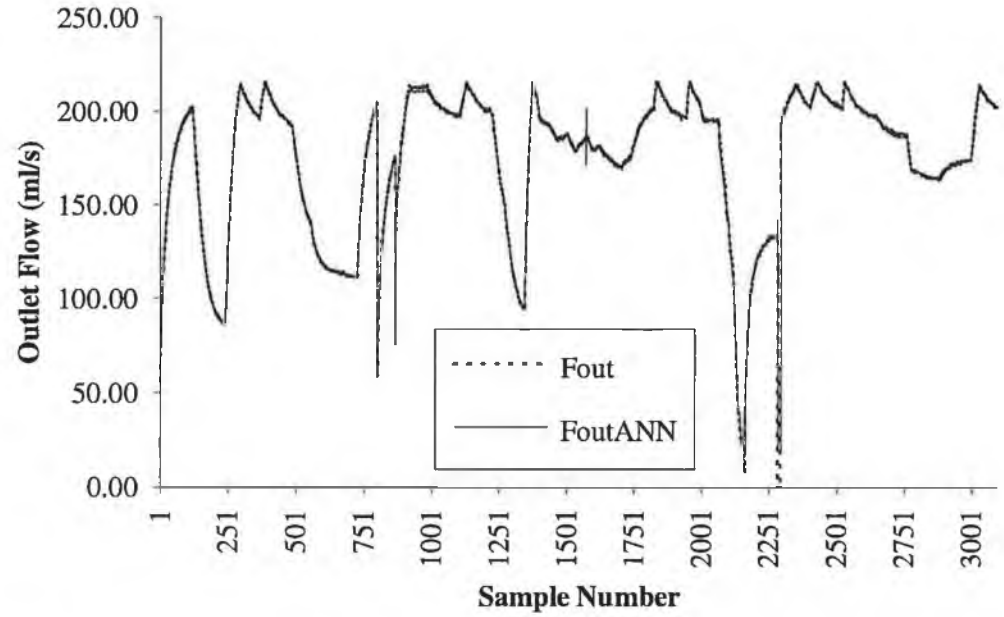


Figure 4.28 - ANN model response to $F_{out}(t)$ training data.

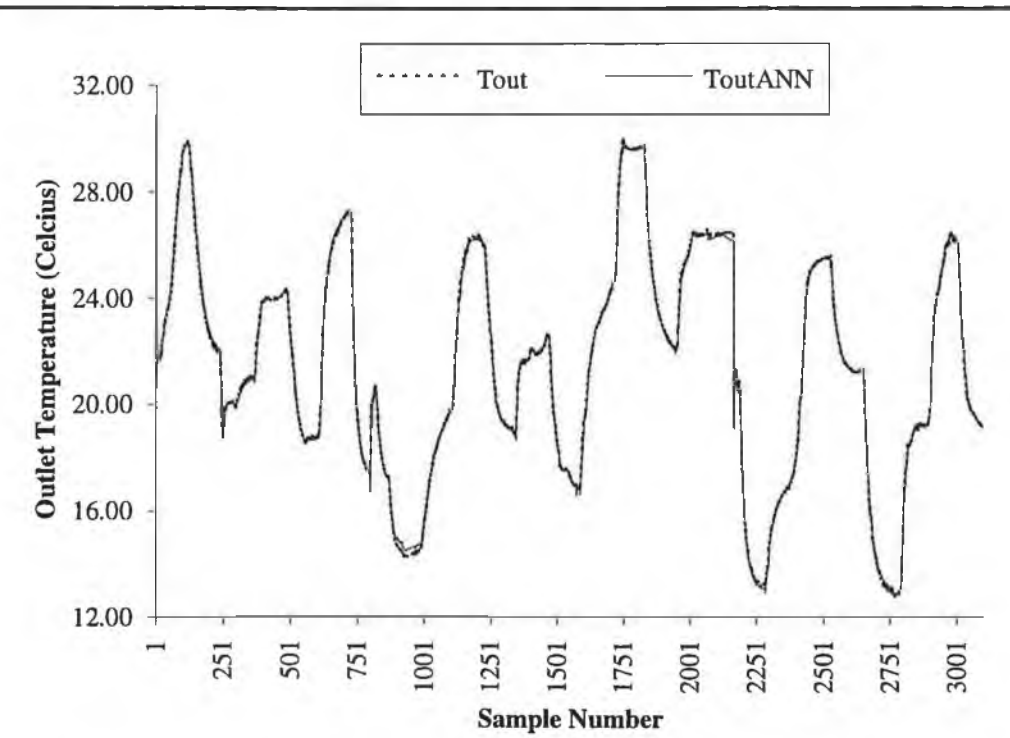


Figure 4.29 - ANN model response to $T_{out}(t)$ training data.

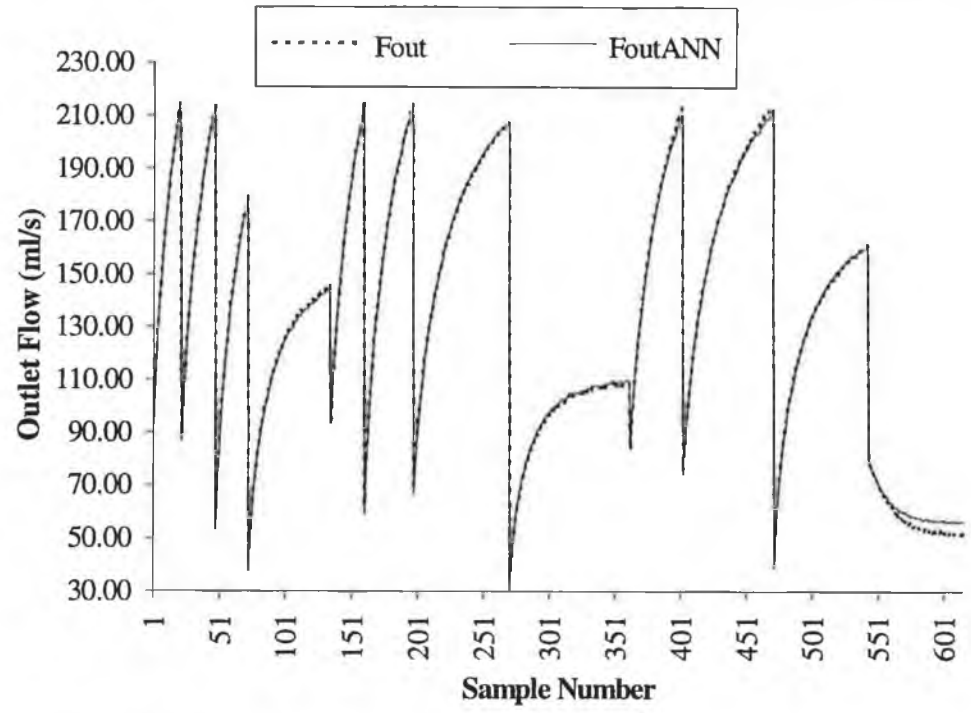


Figure 4.30 - ANN model response to $F_{out}(t)$ test data.

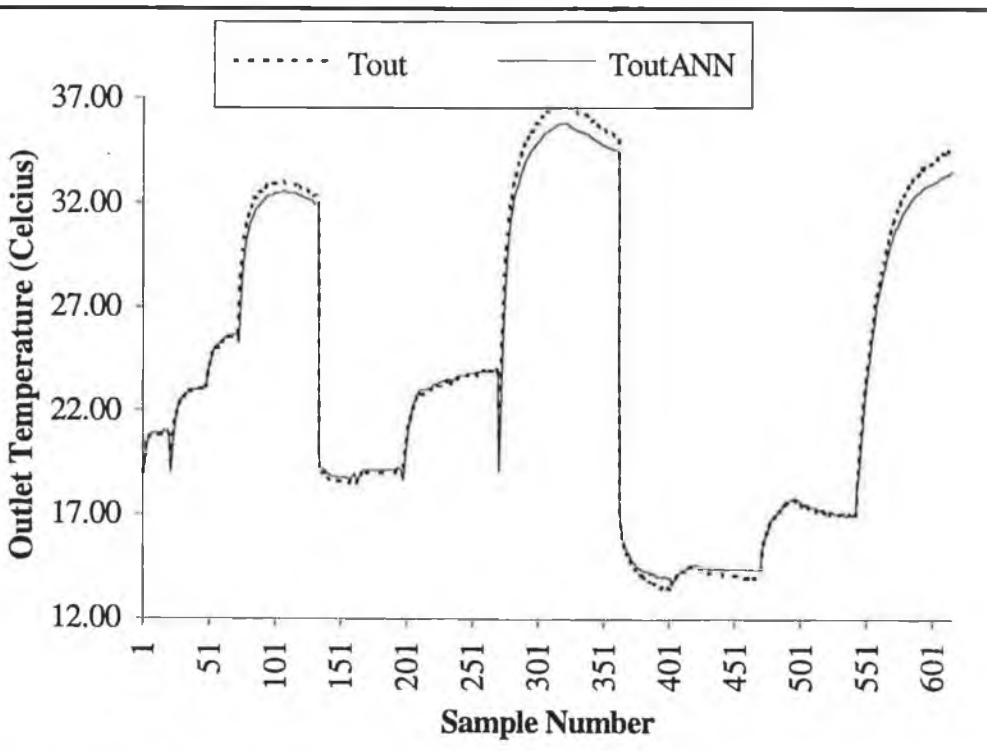


Figure 4.31 - ANN model response to $T_{out}(t)$ test data.

Figures 4.28 and 4.29 (see page 85) show the response of the ANN model of the warm water process to the training data. In order to investigate the generality of the

ANN model of the warm water process, the data contained in the set of step tests listed in Table 4.4 (see page 81) was utilised. This test data set was not included in the training set and thus the ANN model had no previous experience of the data. Figures 4.30 and 4.31 (see page 86) illustrate the response of the ANN model of the warm water process to this test data set. Subjectively it can be concluded that the ANN model is a good one step ahead model of the warm water process. On comparison with the physical model as described in Section 4.5, the warm water process dynamics are represented very well with small offset errors evident in the temperature test data set. The RMS modelling errors for the test data set are 1.73 ml/s and 0.54°Celsius for the outlet flow and temperature variables respectively. These results are better than the best modelling error obtained from the physical model of the warm water process. The results obtained from this architecture and set of data for the ANN model of the warm water process were deemed to be quite sufficient and thus no further ANN architectures were investigated.

4.7. MATLAB/SIMULINK Implementation of the Warm Water Process ANN Model

The *MATLAB Artificial Neural Network (ANN) Toolbox v.1.* has no direct link to the SIMULINK environment [106]. Thus, the ANN model of the warm water process was implemented as a SIMULINK icon manually. This was achieved by creating an *s-function* for SIMULINK within which the ANN output is calculated. The inputs for this icon are the hot and cold inlet flows and the temperature of the hot inlet flow. This ANN Model icon functions only within MATLAB/SIMULINK when the MATLAB Neural Network Toolbox is installed. The trained weights of the network must also be loaded into the workspace and declared as global variables. The macro code used for this icon is contained in Appendix D. Two models were thus available to the user for simulation of the warm water process within the MATLAB/SIMULINK environment. The integration of both of these models into the SIMULINK environment is illustrated in Figure 4.32 (see page 87). As the ANN

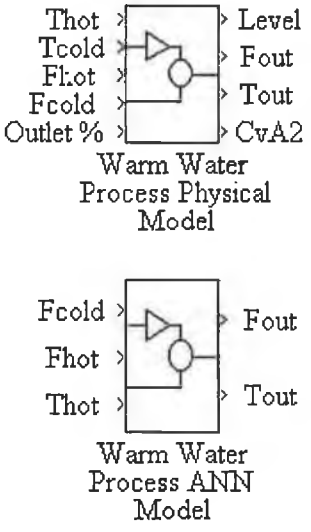


Figure 4.32 - Warm Water Process models in the MATLAB/SIMULINK environment.

model was the most accurate model of the warm water process, it was used in all further simulations.

4.8. Linear System Identification using the ANN Model of the Warm Water Process

In order to gain more insight into the nature of the warm water process and to calculate initial values for the RLS parameter estimator contained in the self-tuning PID controller in Chapter 5, linear system identification was performed on the ANN model of the warm water process. The simulation shown in Figure 4.33 (see page 89) was programmed in order to acquire data from the ANN model of the warm water process for linear identification purposes. Using the MATLAB System Identification Toolbox [107], ARX models for the relationship between the hot and cold inlet flows and the outlet flow and temperature variables of the warm water process were identified. The *general modelling equation* for these relationships is given by the equation (4.24). The *general ARX equation* is given for an input $x(q)$ and an output $y(q)$ in (4.23). $B(q)$ and $A(q)$ are polynomials in q of order n where nb and na are their respective orders. The value nk is the number of steps of pure delay between the input and the output of the system.

$$\frac{y(q)}{x(q)} = \frac{B(q)}{A(q)} q^{-nk} \tag{4.23}$$

$$\begin{bmatrix} F_{out}(q) \\ T_{out}(q) \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} \begin{bmatrix} F_{cold}(q) \\ F_{hot}(q) \end{bmatrix}$$

$$\begin{bmatrix} F_{out}(q) \\ T_{out}(q) \end{bmatrix} = \begin{bmatrix} \frac{B_{11}}{A_{11}} q^{-nk_{11}} & \frac{B_{12}}{A_{12}} q^{-nk_{12}} \\ \frac{B_{21}}{A_{21}} q^{-nk_{21}} & \frac{B_{22}}{A_{22}} q^{-nk_{22}} \end{bmatrix} \begin{bmatrix} F_{cold}(q) \\ F_{hot}(q) \end{bmatrix} \tag{4.24}$$

where M_{11}, M_{12}, M_{21} and M_{22} are ARX models.

Within the MATLAB SYSTEM IDENTIFICATION TOOLBOX, a least squares algorithm is used to fit parameters to ARX models of various orders and delays. Having calculated these parameters for a set of models, loss functions can then be calculated for each model. The user then decides which model structure is most suitable by analysing the loss function values and the standard deviations of the identified parameters.

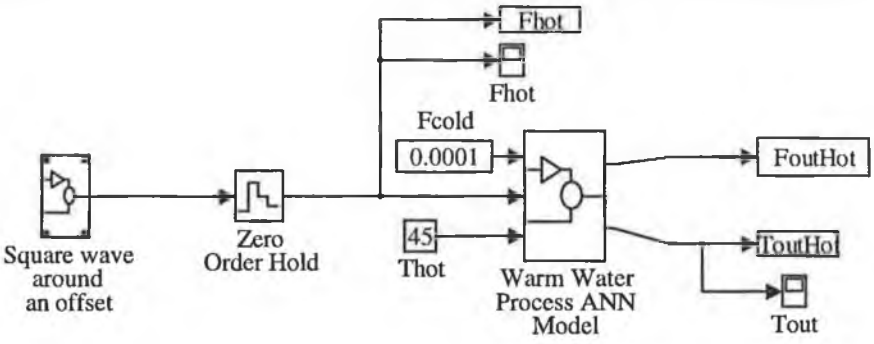


Figure 4.33 - SIMULINK simulation for data for linear system identification.

Table 4.5 - Identified ARX models of the warm water process.

Name	*Output*	*Variable Input*	*Fixed Input*	*A(q) with Standard Deviation*	*B(q) with Standard Deviation*	*Delay*
M_{11} M_{12}	$F_{out}(q)$	$F_{hot}(q)$ 100ml/s ±20ml/s	$F_{cold}(q$ 100ml/s	[1 -9.8271e-1] SD - 1.7361e-4	[0 0 8.3177e-3] SD - 7.1174e-5	2
M_{21}	$T_{out}(q)$	$F_{cold}(q)$ 100ml/s ±20ml/s	$F_{hot}(q)$ 100ml/s	[1 -0.98949] SD - 9.8443e-5	[0 0 -203.55] SD - 1.6218	3
M_{22}	$T_{out}(q)$	$F_{hot}(q)$ 100ml/s ±20ml/s	$F_{cold}(q)$ 100ml/s	[1 -0.9841] SD - 5.9647e-5	[0 0 252.14] SD - 8.567e-1	3

The operating points, structures and parameters of the identified ARX models are detailed in Table 4.5. All of the models have been identified around an inlet flow rate of 200 ml/s. As the ANN model and the physical model of the warm water process are first order, only first order linear ARX models were identified. The models for the relationship between both inlet flows and the outlet flow are identical as the outlet flow is assumed to be independent of temperature. The parameters from these linear models are used in Chapter 5 as initial values for the ARX models contained within the self-tuning PI controllers. This helps to ensure fast convergence of the PI controller parameters and thus ensure reasonable initial control of the plant. Further linear models of the ANN model of the warm water process could be identified for other operating points. These could then be used in the simulation environment to assist the design of other controllers where mathematical models of the plant to be controlled are necessary.

4.9. Fuzzy Modelling

4.9.1. Introduction

The second soft computing method utilised to model the warm water process was *fuzzy modelling*. As in the case of the ANN, the fuzzy model (FM) is another form of vector mapping modelling. Whereas the ANN embodies a non-structured highly non-linear knowledge base, the fuzzy model is characterised by its structured storage of acquired knowledge. The advantage of this structured representation is offset by the increased storage and processing requirements of the fuzzy model. All terminology related to fuzzy modelling used in this section has been defined in Chapter 2.

Generally. the fuzzy model can be characterised by the following points [108] :

- *Non-linear modelling capability* - the fuzzy model is capable of modelling non-linear systems either on-line or off-line.

- *Black box model* - little *a priori* knowledge of the system is required in order to construct a fuzzy model.

- *Linguistic interpretation* - the input and output variables of a fuzzy model can be interpreted linguistically. This allows the initialisation of a fuzzy model with knowledge obtained from a human expert and allows the interpretation of the fuzzy model by a human expert after adaptation, thus performing a teaching function.

- *Embedding in a Controller* - there are several paradigms available for the design of a controller incorporating a fuzzy model [109].

- *On-line adaptation* - due to the structured form of the fuzzy model, a complete training set is not required for each on-line adaptation. In contrast, because the ANN is non-structured a sampled data vector is added to the old training set and the ANN is retrained with this updated training set. The fuzzy model can be trained using only the current sampled data vector and will forget only the stored data contained in the part of its structure that corresponds to the new sampled data vector. This allows fast and convergent learning, making the fuzzy model suitable for on-line learning problems.

There are two main forms of fuzzy models:

- *Scalar consequents* - the consequents of the rulebase are either fuzzy membership sets or fuzzy singletons (crisp values).

- *Function consequents* - the consequents of the fuzzy model are functions - usually of the antecedent variables of the fuzzy model.

As this research is concerned with on-line adaptive fuzzy control, only fuzzy models with scalar consequents are considered. The reason for this is that the adaptation of a scalar consequent is computationally simpler and thus faster than that of a function consequent, which may contain numerous parameters. From the two main types of fuzzy model rulebases for scalar consequent rulebases (the lookup table and relational matrix forms- see Chapter 2), the lookup table approach has been chosen for all fuzzy models in this research, as it requires less memory and is more transparent.

Section 4.9.2 of this chapter details the concept of supervised adaptive fuzzy modelling which is developed within this thesis. In addition, issues pertaining to the architecture and learning mechanism of *Supervised Adaptive Fuzzy Models (SAFM)* are discussed. The development and results of the SAFMs for both the outlet temperature and flow variables of the warm water process are depicted in Sections 4.9.3 and 4.9.4.

4.9.2. Supervised Adaptive Fuzzy Modelling

This section describes the development of a variation of fuzzy modelling that has been termed supervised adaptive fuzzy modelling. This method allows the on-line adaptation of a fuzzy model while guaranteeing convergence for a rule base cell. This guarantee is achieved through the fact that the fuzzy model is only adapted if the modelling error of the fuzzy model will be reduced by the adaptation. Firstly, general issues of fuzzy modelling are discussed by considering each structural element of the fuzzy model in turn as follows :

- *Fuzzification* - the type and distribution of membership sets for each of the antecedent variables

- *Inference* - the choice of the inference operator for linguistic operations such as OR or AND.

- *Rulebase Structure* - the choice of the rulebase structure greatly influences the accuracy and memory requirements of the fuzzy model.

- *Defuzzification* - influences the accuracy and processing time of the fuzzy model.

Thereafter, the learning algorithm used for the supervised adaptive fuzzy model (SAFM) is described. Following this, simulation results of a fuzzy model for a simple first order system are presented and some conclusions are drawn.

4.9.2.1. Fuzzification

Within this section, the *number, type and distribution of membership sets for each antecedent variable* are considered. The more membership sets allocated to each antecedent variable, the finer the resolution and the better the accuracy of the fuzzy model. There are, however, two main disadvantages when the number of sets for each antecedent variable is increased:

- *Large Memory Requirements* - the larger the number of membership sets per antecedent variable, the more memory is required. For a four input fuzzy model with five membership sets per antecedent variable (the rulebase is lookup table form) 5^4 i.e. 625 storage elements are necessary. If we double the number of membership sets for each antecedent variable we then require 10^4 i.e. 10000 storage elements for the rulebase.

- *Large number of Training Data* - by increasing the number of fuzzy membership sets for a particular antecedent variable, the effect of each rule is reduced and thus more training data is required to fill the rulebase.

The type of fuzzy membership set implies whether, for example, triangular, trapezoidal or any other function that fulfils the mathematical requirements for fuzzy membership sets is utilised - see Zimmerman [2]. One popular function used for fuzzy membership sets is the Gaussian function. This function, although requiring more processing time compared with, for example, the triangular set, is differentiable. It is precisely this differentiability which makes this function interesting for adaptive fuzzy models. The advantage lies in the fact that many optimisation algorithms require a differentiable function if they are to perform correctly. The use of the Gaussian function in combination with differentiable functions for linguistic functions e.g. product for AND, allows the use of an optimisation algorithm to optimise some aspect of the fuzzy model e.g. the antecedent variable membership set parameters [15].

The distribution of the fuzzy membership sets for a particular antecedent variable refers to the position of each fuzzy membership set on the universe of discourse. Off-line fuzzy models are often optimised for a particular set of training data by adjusting the position of the antecedent fuzzy membership sets [61]. This research is, however, concerned with on-line fuzzy modelling, where future data can contradict a previously off-line optimised fuzzy membership set distribution. Thus, an equally spaced distribution of fuzzy membership sets for every antecedent variable has been adopted and is maintained, with only adaptation of the centre positions of the consequent fuzzy membership sets being undertaken.

4.9.2.2. Inference

The inference operators used in a fuzzy model implement *linguistic logical concepts* such as *AND or OR*. There are many possible functions available for this implementation - see Chapter 2. There are three main criteria for the choice of the function : *modelling accuracy, processing speed and differentiability*. As the processing requirements of fuzzy algorithms are high, the processing speed of each element of the algorithm is critical. If some form of optimisation algorithm is to be applied to the fuzzy algorithm then the inference function must be differentiable - as described in the Section 4.9.2.1. The functions considered within this research for use in fuzzy modelling and some of there characteristics are listed in Table 4.6. The fastest functions are clearly product and sum, both of which are differentiable.

Table 4.6 - Table of inference functions considered for fuzzy modelling.

	Inference Operators						
	MIN	***MAX***	***Product***	***Sum***	***Fuzzy AND***	***Fuzzy OR***	***Mean***
Linguistic Function	AND	OR	AND	OR	AND	OR	AND /OR
Differ.	No	No	Yes	Yes	No	No	Yes
Speed	Med.	Med.	Fast	Fast	Slow	Slow	Slow

4.9.2.3. Rulebase Structure

There are two types of rulebase structures that can be utilised for fuzzy models with scalar consequents :

- *Lookup Table Format* - only the antecedent variables are used to index to scalar consequent values stored in a lookup table format.

- *Relational Matrix Format* - both antecedent and consequent variables are used to index a possibility value for a rule.

More detailed descriptions of these terms can be found in Chapter 2. The use of the relational matrix format requires considerably more storage space than the lookup table format. The accuracy of the lookup table approach is however, only marginally worse than that of the relational matrix approach [65]. Thus, this research utilises the lookup table format for all rulebases within the fuzzy models subsequently developed.

4.9.2.4. Defuzzification

Defuzzification methods are, to a large extent, dependent on the type of consequents used within a fuzzy rulebase. For the fuzzy models in this research, only centre of gravity methods for either *fuzzy singletons* or *fuzzy variable consequents* are utilised. The fuzzy singleton consequent approach offers superior processing speed. The modelling accuracy of the two approaches is compared in Section 4.9.4.

4.9.2.5. Learning Algorithm

The algorithm used for adaptation of the scalar consequents S is given by (4.25) which has the form of a first order difference equation.

$$\begin{aligned} &if\ \mu_Y \geq \delta \\ &\qquad S_{new} = (1-\alpha)S_{old} + \alpha\,\mu_Y \boldsymbol{Y} \end{aligned} \tag{4.25}$$

The variable $\boldsymbol{Y}$ is the sampled value of the variable to be modelled, e.g. the outlet flow or outlet temperature of the warm water process. Due to the consequent variable vector $\boldsymbol{X}$, several rules will be activated. S_{old} and S_{new} are the old and new values of the scalar consequents of one of the activated rules from the rulebase and μ_Y is the corresponding degree of activation. The learning gain parameter α effects the robustness and adaptability of the fuzzy model. The parameter δ effects the *fuzziness* of the fuzzy model and the inequality $\mu_Y \geq \delta$ corresponds to an alpha-cut as described in Chapter 2. In order to avoid "crosstalk" between cells in the rulebase during adaptation, δ should be not be less than the membership value of the intersection of neighbouring fuzzy membership sets. The higher the value of δ, the less fuzzy the model - if δ is set to unity, then only rules with an activation value of unity can be adapted. The effect of the value of δ is shown is Figure 4.34.

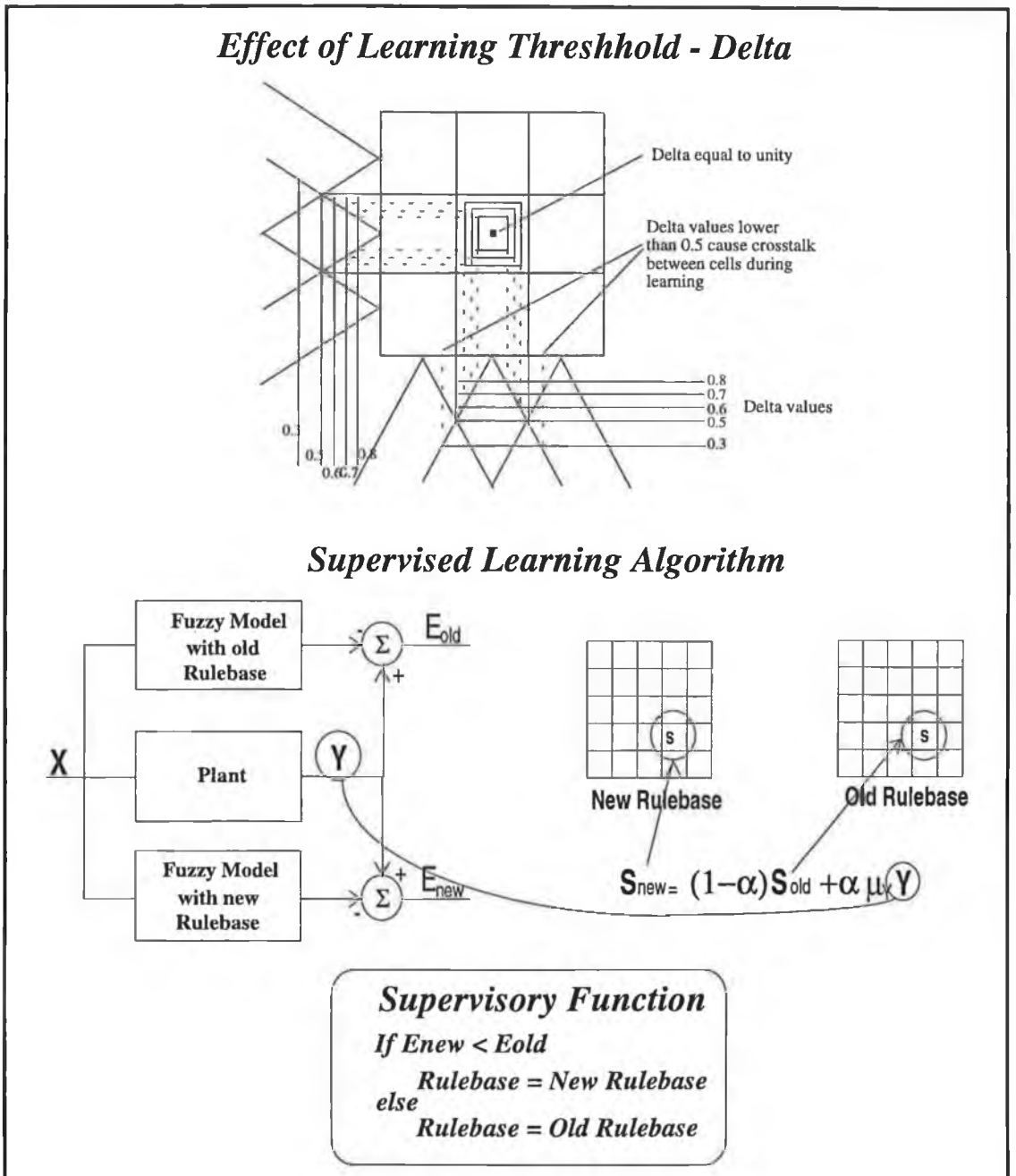


Figure 4.34 - Supervised learning algorithm used for the fuzzy model.

In order to guarantee convergence of the fuzzy model for a particular set of data, the adapted rulebase is only adopted as the new rulebase when it reduces the modelling error of the fuzzy model. This supervisory function leads to the term *Supervised Adaptive Fuzzy Model (SAFM)* and is illustrated in Figure 4.34. The improved modelling accuracy generated by this supervisory mechanism is revealed in the simulations in Section 4.9.3 which develop supervised adaptive fuzzy models for a first order system and investigate the effect of different fuzzy membership sets, inference operators etc. on the modelling accuracy.

4.9.3. Fuzzy Modelling of a First Order System

In order to evaluate the effect of different fuzzy membership set functions, inference operators and the whether singleton or fuzzy variable consequents were to be implemented, a set of fuzzy models of a *first order system* were trained in the MATLAB/SIMULINK environment. The first order system was chosen as it represents a simple and easily understood dynamic system. All models utilise the fuzzy model architecture and learning method discussed in Section 4.9.2. Figure 4.35 shows the SIMULINK simulation used to create these models. The antecedent variables were *u(n-1)* and *y(n-1)*, with the variable *y(n)* estimated by the model.

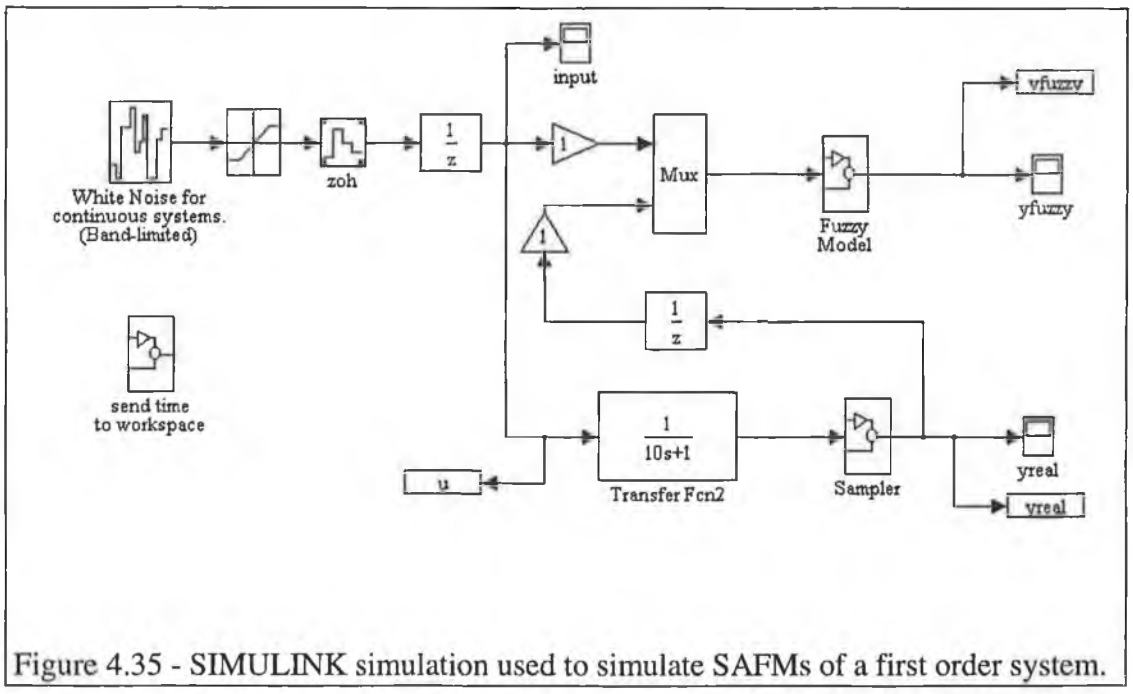


Figure 4.35 - SIMULINK simulation used to simulate SAFMs of a first order system.

In order to train models of the first order system, the plant is driven by a *random signal generator* which changes its output every 5τ seconds, where τ is the time constant of the first order system to be modelled. The training is performed for 500τ so that the state space of the first order system is well covered and the rulebase of the fuzzy model is more or less full, thus forming a global model of the plant. The *number of antecedent fuzzy membership sets* used is *thirteen* unless otherwise specified. This value of thirteen is a good compromise between model resolution and memory requirements. This results in a two dimensional matrix rulebase with 169 elements. As initial values for the rulebase, all 169 cells were set to zero. The value of α in the learning algorithm (4.25) was set to 0.9 and δ was set to a value of 0.5 for all fuzzy models described in this section. Due to the asymptotic nature of the Gaussian function, all the antecedent fuzzy membership sets have a degree of membership greater than zero for a given input value, which slows the fuzzy algorithm considerably. Thus an alpha cut was applied at the fuzzification stage of all

antecedent variables. An alpha cut set to zero and degrees of membership that are less than a user defined threshold. The value of the alpha cut was set to 0.1 thus resetting any degrees of membership less than 0.1 to zero. This alpha cut was applied to all types of antecedent fuzzy membership sets in this investigation.

Six fuzzy models were first trained in order to gain some insight into whether triangular or Gaussian fuzzy membership sets offer better modelling accuracy. Three inference functions - *minimum*, *maximum* and *mean* were used. The results of these simulations are listed in Table 4.7. Clearly the fuzzy models with Gaussian membership sets offer the best accuracy for all three inference operators. Considering these results and the fact that the Gaussian function is differentiable, the Gaussian function was adopted for all future fuzzy models.

Table 4.7 - Prediction performance of recursive fuzzy models (single-step).

Model Number	***Inference Method***	***Antecedent Fuzzy Membership Set Shape***	$\overline{error^2}$
Model 1	Minimum	Triangular	0.1075
Model 2	Mean	Triangular	0.1167
Model 3	Maximum	Triangular	0.0726
Model 4	Minimum	Gaussian	0.0972
Model 5	Mean	Gaussian	0.1075
Model 6	Maximum	Gaussian	0.0421

Table 4.8 - Prediction performance of recursive fuzzy models (multi-step).

Model Number	***Inference Method***	***Antecedent Set Shape***	$\overline{error^2}$
Model 7	Minimum	Triangular	0.1195
Model 8	Mean	Triangular	0.0995
Model 9	Maximum	Triangular	0.0845
Model 10	Minimum	Gaussian	0.0851
Model 11	Mean	Gaussian	0.0908
Model 12	Maximum	Gaussian	0.0666

For comparison purposes, the models detailed in Table 4.7 were again trained but the antecedent variable, *y(n-1)*, was fedback from the fuzzy model output, thus giving a multi-step predictor form. The results of this modification fulfilled the expectations that the modelling performance would not, in general, be as good as the single step predictor form. Table 4.8 (see page 97) contains the results of this investigation.

The following conclusions can be drawn from the results contained in Tables 4.7 and 4.8:

- the *past data y(n-1)* acquired from the plant generally results in better modelling - single step predictor form.

- *Gaussian shaped fuzzy membership sets* for the antecedent variable result in lower modelling errors.

- The *maximum inference function* gives the best modelling performance.

Some cells of the trained rulebase may contain a zero value if no training data influenced these cells of the rulebase. Due to the fuzzy inference mechanism these values will effect the output of the fuzzy model when the input variables approach the neighbouring rulebase cells. Some form of rulebase initialisation can be performed prior to training in order to prevent null value cells from disturbing the output of the fuzzy model.

Fuzzy models 1 to 12, as described in Tables 4.7 and 4.8, utilise *fuzzy singletons as output variable consequents*. In order to evaluate the modelling accuracy achieved by the use of fuzzy singletons as consequent variables, compared to that achieved with fuzzy membership sets as consequent variables, several models utilising a linear distribution of fuzzy membership sets as consequent variables were constructed. Each model used the centre of gravity defuzzification with maximum-product consequent inferencing, resulting in smooth interpolation between the consequent sets. As triangular fuzzy membership sets result in a linear interpolation when combined with the defuzzification method used, these were used for all but one of the fuzzy models in Table 4.9. The number of consequent fuzzy membership sets directly effects the speed of the defuzzification algorithm. Thus the number of consequent sets investigated started at a number slightly less than that of number of antecedent fuzzy membership sets and for comparison purposes finished at a number equal to slightly more than three times this number. The models were tested with the trained rulebase from Model 6 as this gave the best prediction performance from all the previous models. Table 4.9 contains the results obtained.

Table 4.9 - Prediction performance with fuzzy membership sets as consequent variables (single-step).

Model Number	*Consequent Sets*	*Consequent Set Shape*	$\overline{error^2}$
Model 13	9	Triangular	0.0565
Model 14	13	Triangular	0.0551
Model 15	15	Triangular	0.0534
Model 16	19	Triangular	0.0517
Model 17	41	Triangular	0.0456
Model 18	41	Gaussian	0.0445

In conclusion, none of the models using fuzzy membership sets as consequent variables exhibited modelling performance which surpassed that of the fuzzy model Model 6, with fuzzy singleton consequents. Moreover, the time required to calculate the crisp output value using the centre of gravity defuzzification method for fuzzy membership set consequent variables is considerably greater than the fuzzy singleton centre of gravity defuzzification method.

The adaptation applied to the training of the previous eighteen fuzzy models of the first order system was unsupervised. To determine whether the supervisory function, as detailed in Section 4.9.2.5, improves modelling accuracy, several fuzzy models of the first order system were trained using the supervisory learning method. These models were single step predictors, with Gaussian fuzzy membership sets for the antecedent variables and a rulebase consisting of fuzzy singletons as consequents. The results obtained from these simulations are summarised in Table 4.10. When compared with corresponding results from Table 4.7 (see page 97), in the histogram in Figure 4.36 (see page 100), it can be observed that the supervised learning mechanism considerably improves modelling accuracy for all inference functions.

Table 4.10 - Prediction errors using the supervisory learning method (single step).

Model Number	*Inference Function*	*Antecedent Set Shape*	$\overline{error^2}$
Model 19	maximum	Gaussian	0.0286
Model 20	mean	Gaussian	0.0482
Model 21	minimum	Gaussian	0.0522

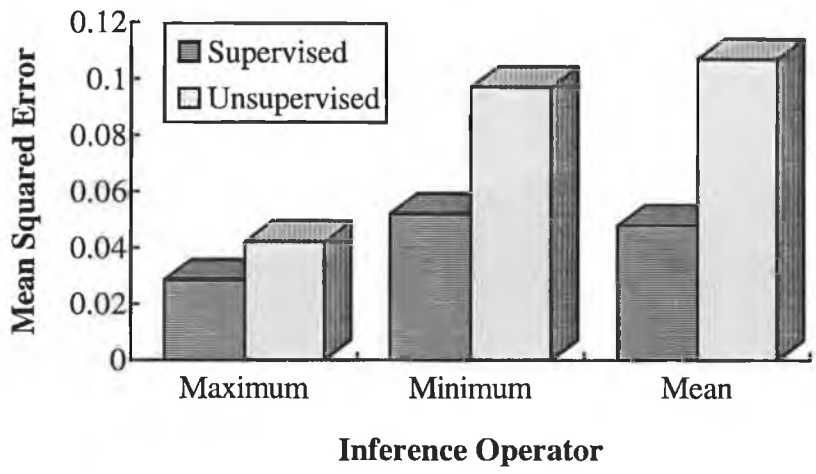


Figure 4.36 - Prediction performance for supervised and unsupervised learning.

Based on the simulation results in this section, *future fuzzy models* using the architecture described in Sections 4.9.2.1 to 4.9.2.5 should have the following characteristics for optimal modelling accuracy:

- *Gaussian antecedent membership sets*
- *Fuzzy singletons as consequent variables.*
- *Supervised learning algorithm.*

The question remains as to whether the conclusions drawn from the results of fuzzy models for linear systems can be extended to the modelling of non-linear systems. The following sections describe the fuzzy modelling of the mass flow and thermal behaviour of the warm water process. The results obtained suggest that the conclusions drawn in this section are applicable to the fuzzy modelling of the warm water process.

4.9.4. Fuzzy Model of the Mass Flow of the Warm Water Process

This section is concerned with the development of a fuzzy model for the relationship between the hot and cold inlet flows and the outlet flow of the warm water process. The fuzzy model structure, as detailed in Section 4.9.3, is utilised. Initial tests were performed within the MATLAB/SIMULINK environment, but due to the processing load the training times for the SAFMs were greater than 30 minutes. It was thus decided to develop the necessary software for the training and testing of the SAFMs

in the "C" programming language to enable the training of the rulebases on an IBM compatible PC or a UNIX based workstation, using a C source code implementation of the neural network as a reference plant. Details of this software are given in Chapter 7, which gives an account of the software engineering issues of this research. The result of this recoding was a one hundred fold speed increase when running on an IBM PC 486DX2 50Mhz. The C code was also ported to a SUN SPARC 10 workstation where the speed of the software was again increased.

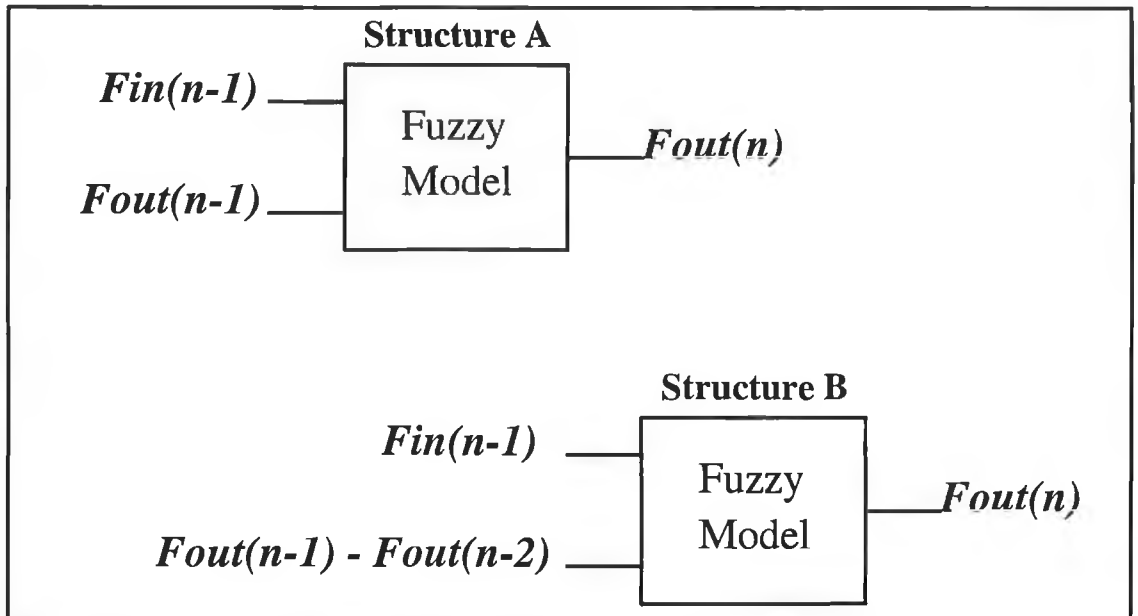


Figure 4.37 - Possible fuzzy model structures for mass flow behaviour.

As the *mass flow behaviour* of the warm water process is first order, the fuzzy model used to model this mass flow behaviour has two inputs which contain the following information :

- *the sum of the two inlet flow into the warm water process* and
- *the previous outlet flow value.*

Two possible structures for the SAFM of the mass flow behaviour of the warm water process with these two inputs are shown in Figure 4.37. $F_{in}(n)$ is the sum of the inlet flow values and $F_{out}(n)$ is the outlet flow of the warm water process. These two structures- *Structure A and Structure B* - differ through the choice of the outlet flow antecedent variables.

Structure A utilises the old value of the outlet flow $F_{out}(n-1)$. This method has the disadvantage that a large number of antecedent fuzzy membership sets are required in order to model the dynamics of the process i.e. the difference between $F_{out}(n)$ and

$F_{out}(n\text{-}1)$. This is because when the change in F_{out} over successive samples is less than the width of one rulebase cell, the same rulebase cell consequent value is adapted and thus old information is lost. As the change between two samples for a system sampled at 0.1τ where τ is the dominant time constant, is small in comparison to the width of the universe of discourse of the outlet flow variable i.e. 240 ml/s, this form of fuzzy model has inferior resolution in comparison to Structure B. For the example shown in Figure 4.38, when the difference between $F_{out}(n\text{-}1)$ and $F_{out}(n\text{-}2)$ is smaller than one rulebase cell (for five cells this corresponds to 20% of the width of the universe of discourse of the outlet flow i.e. 48 ml/s), then information will be lost.

Structure B utilises the difference $F_{out}(n\text{-}1)\text{-}F_{out}(n\text{-}2)$ as an antecedent variable. This approach has better resolution than Structure A and thus offers better modelling accuracy of the plant dynamics with fewer antecedent membership sets. The necessary span of the universe of discourse for this *difference* variable can be approximately calculated by considering the maximum change possible for the value of $F_{out}(n\text{-}1)\text{-}F_{out}(n\text{-}2)$. Assuming a linear first order response and a sampling rate of 0.1τ, the maximum value corresponds to approximately 20% of the maximum value of the input flow i.e. ±80 ml/s. This offers better resolution per rulebase cell than Structure A. Based on these considerations and the results of initial simulations, Structure B was chosen for the fuzzy model of the mass flow behaviour of the warm water process.

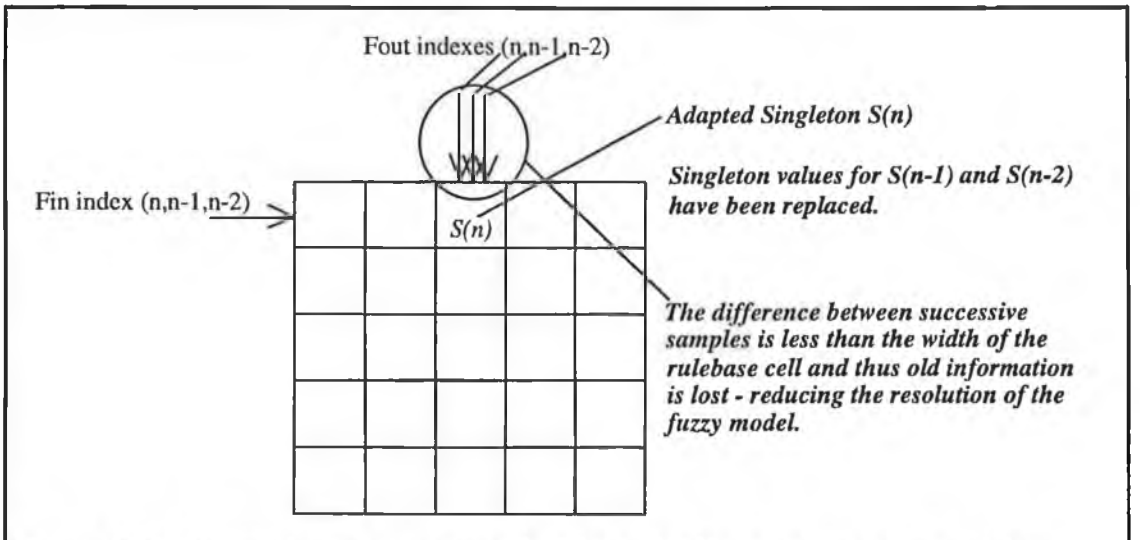


Figure 4.38 - Resolution of fuzzy model with $F_{out}(n\text{-}1)$ as an antecedent variable.

Having decided on Structure B for the fuzzy model of the mass flow behaviour of the warm water process, the rest of the fuzzy model parameters were chosen.

The experience gained in the modelling of the first order system was drawn upon. Thus all antecedent fuzzy membership sets were Gaussian functions, the model is in the form of a single step predictor and supervised learning is used to train the rulebase. For each of the antecedent variables the number of fuzzy membership sets used was twenty one. This was based on previous experience with the fuzzy models of the first order system, where thirteen antecedent sets were used, but as this system is non-linear, slightly more resolution was deemed necessary. This *choice of 21 fuzzy membership sets per antecedent variable* resulted in a *rulebase with 441 (21x21) storage elements*, each of which contained a fuzzy singleton. The linguistic function AND was approximated by the *product* function based on its speed of execution and the ease its of software implementation.

To train the fuzzy model of the mass flow behaviour of the warm water process, the *C source code implementation on a SUN SPARC 10 workstation* was utilised. In order to generate a global model i.e. fill the rulebase, the response of the ANN model to random input flow values were used to train the fuzzy model. Each fuzzy model was trained for the equivalent of 10 months i.e. 20 million seconds corresponding to 5000 random inlet flow values.

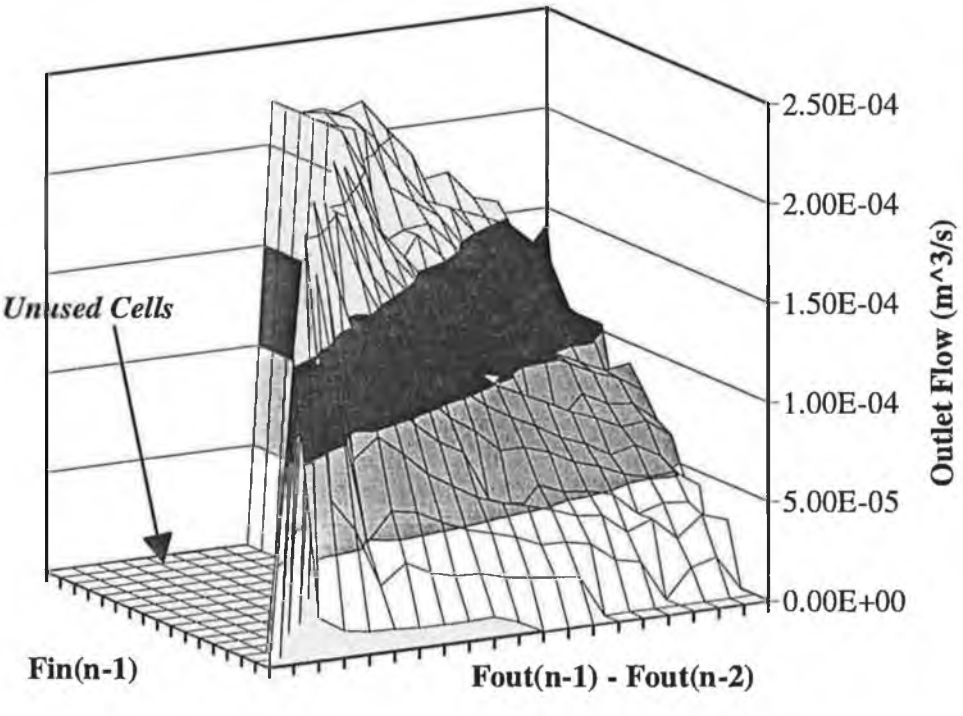


Figure 4.39 - Rulebase characteristic of the fuzzy model of the mass flow behaviour.

The effect of the parameter δ from the learning algorithm (4.25) on modelling accuracy of the SAFM was investigated. This involved evaluating SAFMs trained with various values of δ of between 0.5 (the intersection value of the antecedent fuzzy membership sets) and 1.0. In addition, the effect of an alpha cut for the membership values of the antecedent (described in Section 4.9.3) variables on the performance of the SAFM was investigated. A value of 0.7 for δ and an alpha cut threshold of 0.1 gave the best RMS modelling error.

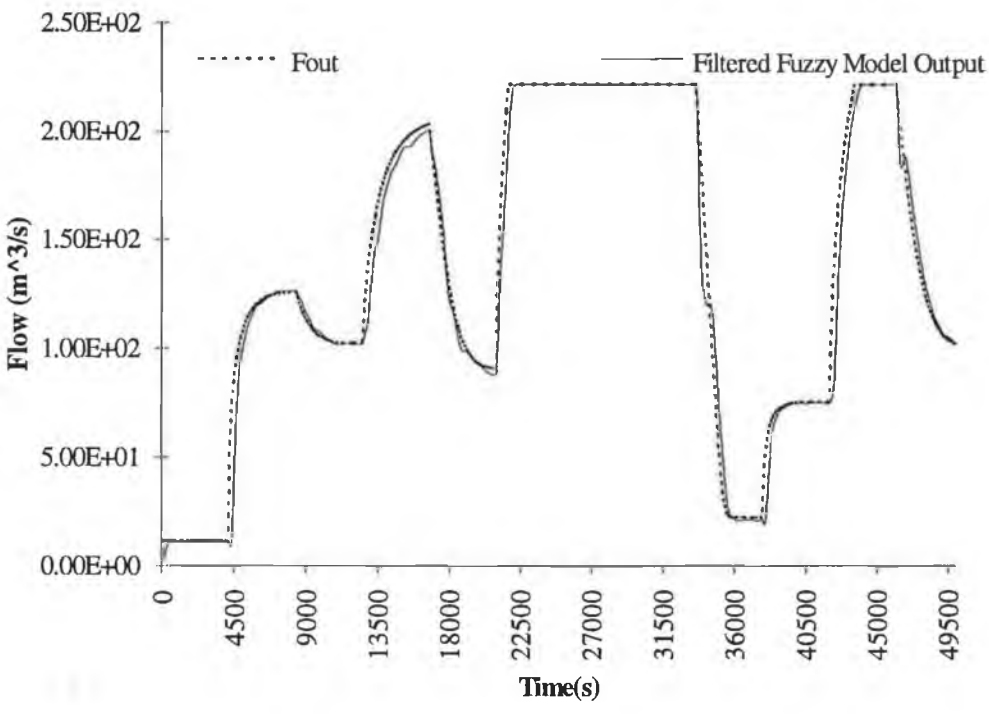


Figure 4.40 - Evaluation of SAFM for mass flow.

Figure 4.39 (see page 103) shows the characteristic of the trained rulebase. A total of 281 from 441 cells are used, corresponding to a usage of 64%. This usage is dependent on the plant dynamics as a full training set is assumed. As untrained cells effect the accuracy of the model when neighbouring cells are indexed, the output of the fuzzy model was filtered, which is equivalent to smoothing the surface of the trained rulebase. A first order low pass Butterworth filter with a time constant 200 seconds was used.

One possible method that would reduce the memory requirements of fuzzy rulebases is the storage of only the used portions of the rulebase - similar to the storage mechanisms used for sparse matrices.

Figure 4.40 shows a comparison between the responses of the SAFM for the outlet flow of the warm water process and the ANN model used to train the SAFM to a signal consisting of ten setpoint changes. The results of this modelling strategy are subjectively good, with the dynamics of the mass flow being modelled quite well.

4.9.5. Fuzzy Model of the Thermal Behaviour of the Warm Water Process

The main problem encountered during the development of the fuzzy model for the thermal behaviour of the warm water process was the *specification of the input variables and the number of their respective fuzzy membership sets.* Due to memory considerations, the number of input variables for the fuzzy model was to be kept as low as possible. Experience gained from the physical and ANN models of the warm water process was used to decide exactly which variables were utilised as inputs. Figure 4.41 (see page 105) shows the structure of the fuzzy model for thermal behaviour and the input variables chosen. These input variables are described by the following :

- *Cold inlet flow* $F_{cold}(n-1)$.
- *Hot inlet enthalpy* $E_{hot}(n-1)$ - is the product of the hot inlet flow and the hot inlet temperature - $F_{hot}(n-1)*T_{hot}(n-1)$. This combines two input values and helps to reduce the dimensions of the fuzzy rulebase.
- *Outlet Flow* $F_{out}(n-1)$ - this variable is directly related to the level of water in the process reaction tank and thus reflects the mixing dynamics of the process.
- *Previous Change in the Outlet Temperature* $T_{out}(n-1)-T_{out}(n-2)$ - choice is based on the same considerations as the choice of Structure B (Figure 4.38, page 104) for the fuzzy model of the mass flow behaviour of the warm water process as described in Section 4.9.3.

This choice of input variables results in a *'four dimensional rulebase.* Because one of the control strategies for the warm water process is one step prediction based on a fuzzy

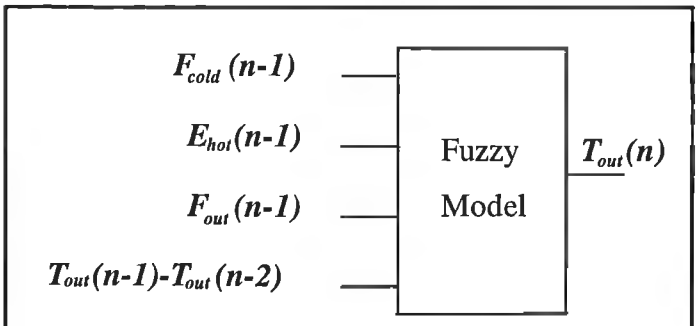


Figure 4.41 - Fuzzy model structure for the warm water process thermal behaviour.

model of the warm water process (see the conclusion to Chapter 2), the memory limitations of the computer used for process control had to be considered during the design of the fuzzy models. The Turbo C compiler used for programming creates DOS programmes which have a maximum memory address space of 640kBytes. Thus, all fuzzy models must be considerably less than 640kBytes in size. The number of cells in the rulebase is calculated by calculating the product of the number of antecedent fuzzy membership sets of the input variables. Each cell contains a short floating point number which requires four bytes for storage. With 7 fuzzy membership sets per antecedent variable, the rulebase required 9604 bytes of memory (7^4*4). With 21 fuzzy membership sets per antecedent variable, the required memory is 777,924 bytes (21^4*4), clearly too large. After detailed simulation, a configuration of *15 Gaussian fuzzy membership sets for each antecedent variable* was decided upon. This has a memory requirement of 202,500 bytes (15^4*4), offering a good compromise between modelling accuracy and storage requirements.

As in the case of the fuzzy model of the mass flow of the warm water process, the rulebase cells contains fuzzy singletons as consequent fuzzy membership sets, all of which are trained using supervised learning. The same values for the learning gain α, the learning threshold δ and the applied alpha cut as the fuzzy model for mass flow behaviour were used i.e. α=0.9, δ=0.7 with an alpha cut at 0.1. The rulebase was trained in the same manner as the mass flow rulebase (described in Section 4.9.3) for 50 million seconds.

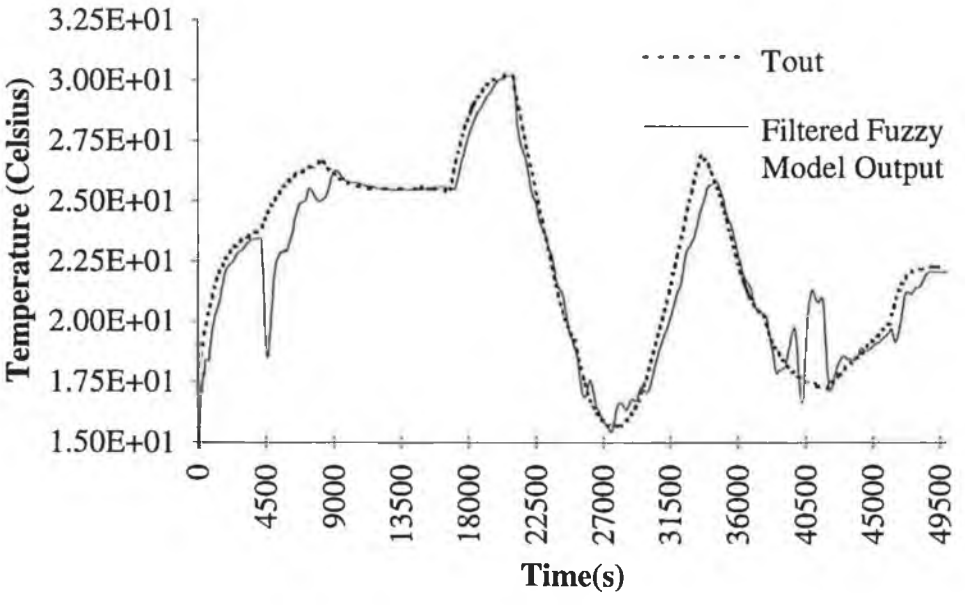


Figure 4.42 - SAFM for thermal behaviour of the warm water process.

A comparison of the output of this SAFM and the ANN model used to train it to a series of random setpoint changes, is illustrated in Figure 4.42. The output of the fuzzy model is filtered with a second order lowpass discrete Butterworth filter. The cut-off frequency of the filter is 8.335×10^{-4} Hz, which was chosen subjectively after analysing the results of several other cut-off frequencies. A second order Butterworth filter was chosen as this has no passband ripple. The modelling capability of the fuzzy model for the thermal behaviour of the warm water process is not as good as that of the mass flow fuzzy model. The sparseness of the rulebase was examined using the *spy* function in MATLAB [110]. From the 50625 available cells in the rulebase, only 18390 were used by the rulebase after training for over 24000 random inlet setpoint values applied to the ANN model of the warm water process. This sparseness is attributable to the ANN model dynamics, which is not a completely global model of the warm water process. Thus considerable gains in memory efficiency can be attained if the rulebase is stored as a sparse matrix.

4.10. Conclusions

The first principles model developed in this chapter, although proving to be a good model for the mass flow of the warm water process, was incapable of modelling the thermal behaviour of the warm water process satisfactorily. This failure is attributable to the highly non-linear characteristic of the outlet temperature which could not be modelled by the physical model which assumes perfect mixing.

Because newer modelling strategies from the artificial intelligence field have been shown to be capable of modelling a system based on a good set of training data, an artificial neural network model of the warm water process was developed. The modelling capability of the ANN model was quite good, with RMS modelling errors, for a set of test data that was not used for training, of less than 2% for both outlet variables. Based on this result the ANN model was implemented in the MATLAB/SIMULINK environment and was used as a simulation model of the warm water process.

Linear system identification was performed on the ANN model of the warm water process to develop first order ARX models around one operating point. Thus more insight was gained into the plant dynamics. Moreover, the identified ARX models serve as initial values for the ARX models of the self-tuning PI controllers which are used for comparison with adaptive fuzzy control strategies in Chapter 5.

Using a first order system as a reference plant, a strategy for adaptive fuzzy modelling was developed which utilises on-line supervised learning. This adaptive fuzzy modelling strategy is termed *Supervised Adaptive Fuzzy Modelling* and allows convergent on-line adaptation of scalar consequent variables of a fuzzy model with a lookup table rulebase. Moreover, it forms an integral part of an adaptive fuzzy control strategy for the warm water process - *Single Step Predictive Fuzzy Control* - which uses fuzzy models of the warm water process. Separate fuzzy models of the mass flow and thermal behaviour of the warm water process were developed. The fuzzy model of warm water process mass flow gave quite good modelling results. The fuzzy model of the thermal behaviour results were not as good as those of the mass flow especially when the storage requirements for the rulebase are considered.

Chapter 5 - *Controller Design*

5.1. Introduction

5.1.1. General Introduction

This chapter describes the design and evaluation of adaptive fuzzy controller methods studied in this thesis. As stated in Chapter 1 and the concluding remarks of Chapter 2, *two strategies* for adaptive fuzzy control of the warm water process have been chosen for further investigation and development :

- *Self-Organising Control* - a direct adaptive fuzzy control algorithm which allows the on-line adaptation of rulebase consequents based on a user defined reference model, which is often of a heuristic nature. Most of the literature found within the field of adaptive fuzzy control is concerned with this controller paradigm.

- *Single-Step Predictive Fuzzy Control* - an indirect adaptive fuzzy control strategy which utilises a fuzzy model to predict the plant behaviour over a single sample. An optimisation strategy then calculates the controller output so that a user defined cost function is satisfied. This controller paradigm is based on that from Moore and Harris [19] with the following contributions made by this research,

    1. the *step function reference model* used by Moore and Harris is *extended* to a *discrete first order system*, thus improving the overall controller response and enabling the user to more fully specify the desired dynamic response of the system,

    2. the *use of the supervised adaptive fuzzy modelling strategy* developed in Section 4.9, thus enabling on-line adaptation, whilst helping to improve fuzzy model convergence and

    3. the *application of the controller to a multivariable control problem.*

Where possible, both of these adaptive fuzzy control methods are simulated in detail for the control of the outlet flow, the outlet temperature and the multivariable control of both the outlet and temperature variables of the warm water process.

For evaluation purposes, a comparison is made between the two adaptive fuzzy control methods and a self-tuning PI controller which is based on an algorithm from

Banyasz and Keviczky [111]. This research contributes the following to the original algorithm :

- the modification of the original self-tuning PID controller algorithm to that of a self-tuning PI controller algorithm and

- the addition of PI controller parameter limits, high pass filtering for identification of systems with DC offsets and an anti-integral-windup mechanism.

The ST-PI algorithm developed is firstly applied to the simulated control of the outlet flow and outlet temperature of the warm water process. Through the implementation of static decoupling of the controlled variables, the multivariable control capability of the ST-PI controller algorithm is investigated.

5.1.2. Overview of Chapter Structure

Section 5.2 of this chapter describes the simulation of the Self-Organising Controller (SOC) for control of the outlet flow of the warm water process. This research combines individual elements of the SOC designs from original SOC literature reviewed in Section 2.7.3.1, in order to attempt to simplify and improve the SOC. *Section 5.3* contains a detailed account of the development and simulation of single step predictive fuzzy controllers for the control of the outlet flow, the outlet temperature and multivariable control of the outlet flow and temperature of the warm water process. *Section 5.4* describes the development of the self-tuning PI (ST-PI) controller algorithm for the control of the outlet flow, the outlet temperature and the multivariable control of the outlet flow and temperature of the warm water process. Finally *Section 5.5* compares and contrasts the three adaptive control strategies that have been simulated and, based on these results, chooses an adaptive fuzzy control strategy for evaluation on the real warm water plant.

5.2. Self-Organising Control

5.2.1. Introduction

This section presents a subset of the results obtained from simulation carried out to investigate the ability of the *Self-Organising Controller (SOC)* to control the warm water process. The structure and development of the SOC since its inception by Procyk and Mamdani in 1979 [17] is described in Section 2.7.3 of this thesis.

The SOC utilises a *reference model* to directly adapt the rulebase consequents based on the current controller performance and is commonly based around the PD-FLC described in Section 2.5.2 of this thesis. Figure 2.4, see page 26, shows the general structure of the SOC for the control of a SISO plant.

5.2.2. Chosen SOC Structure

The SOC used for the simulations in this section is a combination of various aspects of different SOC architectures that were covered by the literature survey. The following points describe in detail, the SOC architecture used and justify the chosen structure and parameters.

- *Input and output variables* - for the control of a SISO system, the majority of SOC designs use the PD-FLC format where the *error* and *change in error* serve as input variables and an incremental value serves as a controller output. As this is the most common configuration found in the SOC literature, it is used for the SOC here.

- K_e, K_{de}, K_u - the input and output variable gains. For the input variable *error* the value of K_e = 16666 was chosen. This value of K_e which maps the maximum error values of $\pm 0.00060 m^3/s$, to the universe of discourse of the fuzzy sets of [-1,1]. For values of error with a magnitude greater than $0.00060 m^3/s$, the controller output saturates at a maximum value. This *error* mapping configuration allows finer dynamic control of the plant for smaller values of error magnitude. The value of K_{de} = 20833 was chosen, as this maps a change of error value of ±0.000048 (20% of the maximum outlet flow) to the fuzzy sets universe of discourse of [-1,1]. The value of the output variable gain, K_u, is varied during simulation in order to evaluate its effect on controller performance.

- *Rulebase format* - the original SOC from Procyk and Mamdani [17], used a relational matrix rulebase. As shown by Wakileh and Gill [50], the relational matrix rulebase form requires considerable processing time and memory resources. In order to avoid this difficulty, a number of SOC implementations, e.g. Ho and Lin [52], Spinrad [54], Burkhardt and Bonnisone [14], utilise the simpler lookup table rulebase format. Based on these considerations, the SOC structure described in this section utilises the lookup table rulebase format, thus increasing the transparency of the SOC.

- *Fuzzy membership sets* - the two antecedent variables, *error* and *change in error*, utilise a regularly spaced distribution of 13 triangular fuzzy membership sets with universes of discourse of [-1,+1], whereas the consequent variables are in fuzzy singleton form. The justification for this antecedent variable format is the fact that it is used in most of the SOC literature, giving a reasonably fine rulebase resolution whilst maintaining good interpolation characteristics. The choice of consequent variable format is based on the increased flexibility of the fuzzy singleton compared to the fuzzy membership set representation, shown in the fuzzy modelling section of this thesis, Section 4.9.3. It should be noted that the choice of the number of fuzzy sets in the SOC is subjective by nature. The choice of the form and number of fuzzy membership sets to be used in a fuzzy controller highlights one of the disadvantages of fuzzy control - *the large of number of parameters to be chosen by the designer.* This disadvantage has been discussed in Section 2.5.1. of this thesis.

- *Reference model* - the reference model (usually known as the "performance index" in the SOC literature) is used by the SOC architecture to evaluate the current controller performance and to adapt the rule consequent values. The reference model uses the *error* and *change in error* variables as inputs (see Figure 5.1). In the original SOC design from Procyk and Mamdani [17], the output of this reference model is used, together with the incremental plant model, to adapt the rulebase consequents. Many of the publications reviewed in Section 2.7.3 utilise a heuristic lookup table for the reference model, whereby little detail is given to its origin. In order to test the characteristics of a heuristic reference model, the reference model detailed by Sugiyama [49], is initially used in these simulations. This reference model is enhanced in this research by converting it from a lookup table to a fuzzy algorithm. This fuzzy algorithm format allows interpolation between its rulebase elements instead of the hard switching between elements, which is characteristic of lookup tables. Regularly spaced distributions of triangular fuzzy membership sets for the antecedent variables, fuzzy singletons as consequent variables, the product inference operator and the centre of gravity defuzzification method are utilised by this fuzzy algorithm, where the lookup table from Sugiyama serves as rulebase consequents. This choice of fuzzy algorithm structure and parameters guarantees linear interpolation between the elements of the rulebase.

- *Rule consequent adaptation* - The lookup table rulebase format enables the replacement of the so-called "incremental process model" [17], with a simple learning law, as detailed by Ho and Lin and given by equation (2.8), see page

29. Through the utilisation of this simple learning algorithm, rulebase learning is dependent on the learning gain α. The effect of the value of the learning gain α from this equation is to be investigated in this section. As detailed in Section 5.7.3, the SOC adapts the consequent values of the past rulebases that have contributed to the current performance of the controller and places the adapted consequent values in the current rulebase. Precisely which past rulebases are to be updated is not specifically described in any of the literature. Some authors claim that the SOC performance is insensitive to which past rulebases are updated [17,49]. This seems unlikely as the effect of the rules responsible for the current controller performance can be assumed to be delayed by a time, t_m, equivalent to the dominant time constant of the plant. Spinrad [54] adapts multiple past rulebases and weights these updates with an "importance weight". In this simulation, the rulebases between 13 and 8 previous samples are adapted with unity weighting. This choice is based on the assumption that the sampling rate of 30 seconds approximately corresponds to one tenth of the dominant time constant of the plant. In addition, no rulebase adaptation is performed after a setpoint change for a period of time equal to the maximum rulebase update delay, i.e. 13 samples. This ensures that past rulebases that were not active for the current setpoint, are not adapted. The MATLAB/SIMULINK block "General RB Modifier", see Figure 5.2 on page 114, is used to update the rulebases. This block allows a weighted update of more than one previous rulebase as suggested by Sugiyama [49] and Spinrad [54] and allows the application of a *zero central value* and *symmetrical update* as suggested by Farbrother, Stacey and Sutton [53]. The value of the magnitude of the rulebase consequents is limited in order to prevent system instability due to large rulebase consequent values, which can be caused by continuous adaptation of a single rulebase cell.

5.2.3. SOC Simulation Software

The *MATLAB/SIMULINK software* used for the SOC simulations in this section is shown in Figures 5.1. and 5.2, on page 114. As an initial study, the outlet flow of the warm water process was controlled in simulation. Because it is a good model of the mass flow behaviour of the warm water process, as shown in Section 4.6, the artificial neural network model of the warm water process is utilised as the plant for these initial simulations. If these simulations proved to be promising, then temperature and multivariable control were to be attempted.

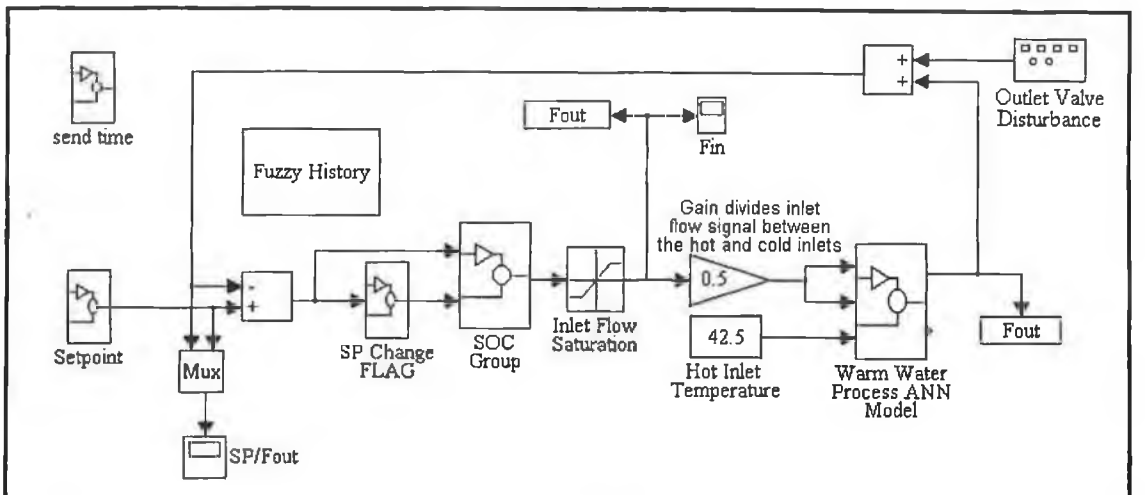


Figure 5.1 - Simulated SOC for warm water process outlet flow control.

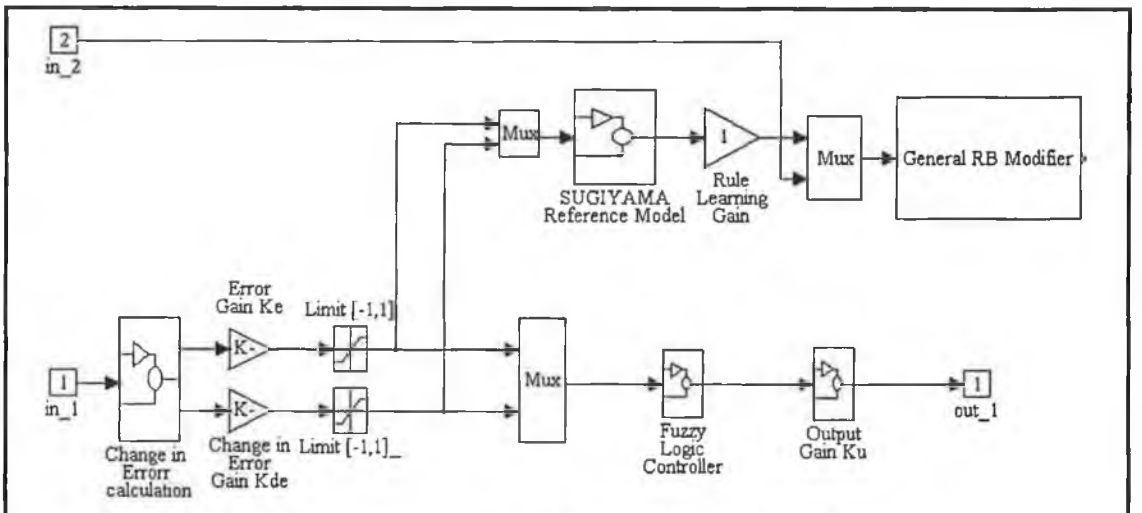


Figure 5.2 - Internal structure of the "SOC Group" from Figure 5.1.

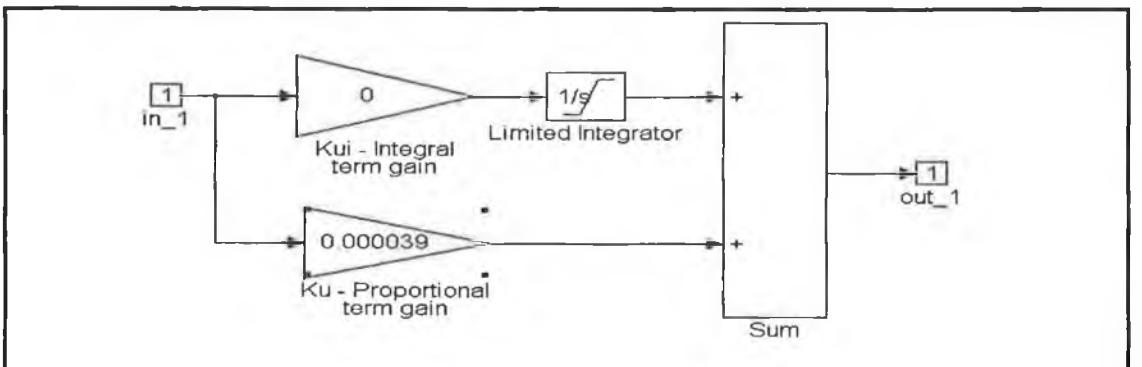


Figure 5.3 - Internal structure of the "Output Gain Ku" block from Figure 5.2.

The SOC used in this simulation has a *large number of parameters* that need to be selected by the user and subjectively optimised during simulation. This large number of parameters proved to be confusing in practice, with many different combinations of parameter values being investigated before the configuration used as a basis in this

section was arrived at. During this process, the values of the learning gain α for rule update and the controller output K_u were seen to be critical for the performance of the SOC. Thus, different values of these two parameters including an integral term in the output gain block K_{ui}, as shown in Figure 5.3, see page 114, are investigated in this section. The purpose of the integral term in the output gain was to attempt to reduce the steady state error of the SOC that was observed during simulation.

5.2.4. Simulation Results

The *results of the first set of simulations are shown in Figures 5.5 to 5.11*, pages 116 and 117. It should be noted that the label F_{in} used in these graphs, is equivalent to the sum of the hot and cold inlet flow. The values of the learning gain α , see equation (2.8) page 29, and the output variable gain are contained in Table 5.1, together with the corresponding test names. All simulations have a setpoint of 100 ml/s, with a square wave disturbance signal with a period of 6283 seconds and an amplitude of 25 ml/s, added to the plant output. This disturbance signal, as shown in Figure 5.4, corresponds to a disturbance on the outlet flow valve actuating signal, which by affecting the outlet valve position, directly influences outlet flow. The SOCs task is to keep the outlet flow at 100 ml/s and to reduce the effects of the disturbance signal to a minimum.

Table 5.1 - Table of parameters for the SOC simulations.

Test Name	**Value of α**	**Training Cycle**	K_u	K_{ui}
SOC Test 1	5	1	0.000039	0
SOC Test 2	0.5	1	0.000039	0
SOC Test 3	1	1	0.000039	0
SOC Test 4	1	2	0.000039	0
SOC Test 5	1	3	0.000039	0
SOC Test 6	1	2	0.000039	0.00000015
SOC Test 7	1	2	0	0.00000015

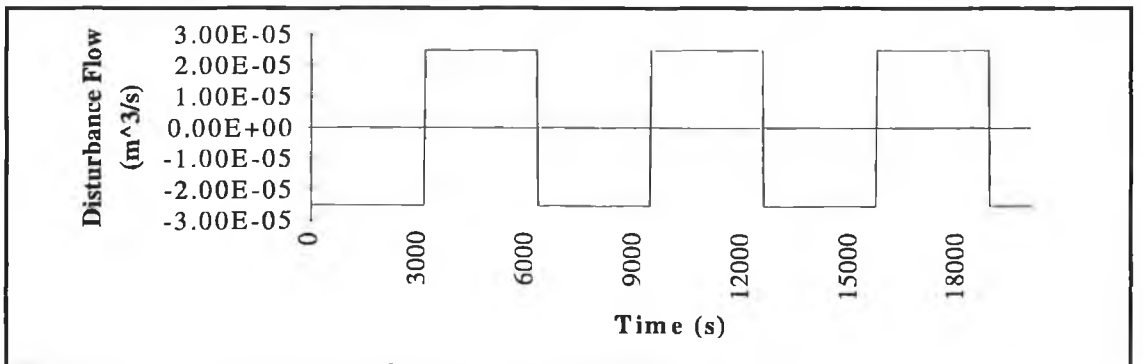


Figure 5.4 - Disturbance on the outlet valve actuating signal.

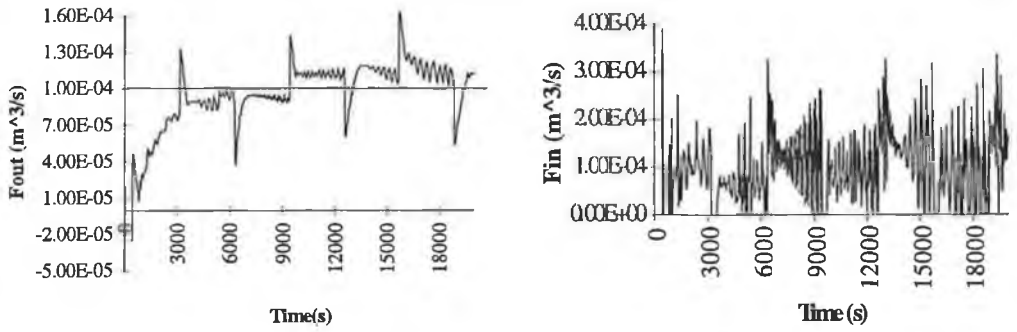


Figure 5.5 - Response of SOC for outlet flow control - SOC Test 1.

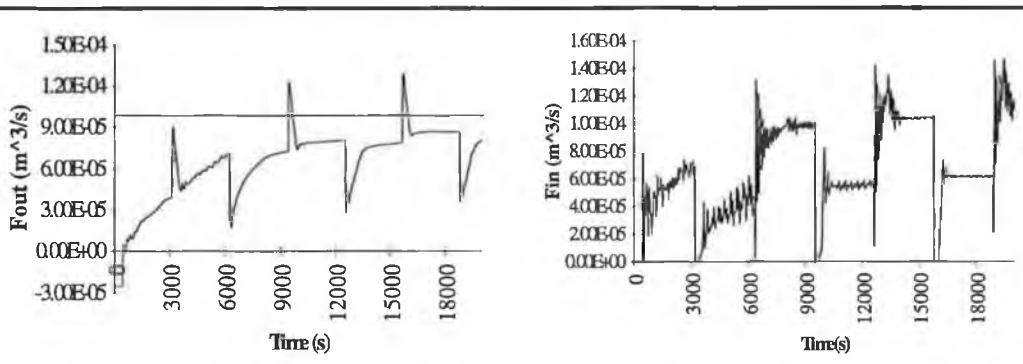


Figure 5.6 - Response of SOC for outlet flow control - SOC Test 2.

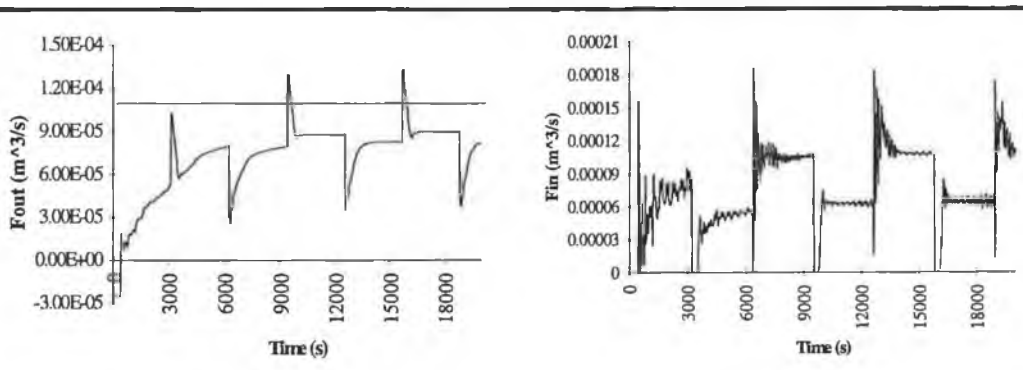


Figure 5.7 - Response of SOC for outlet flow control - SOC Test 3.

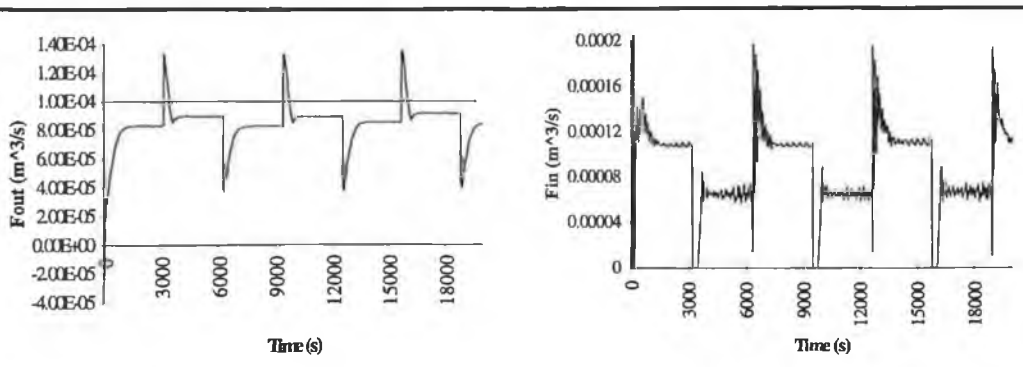


Figure 5.8 - Response of SOC for outlet flow control - SOC Test 4.

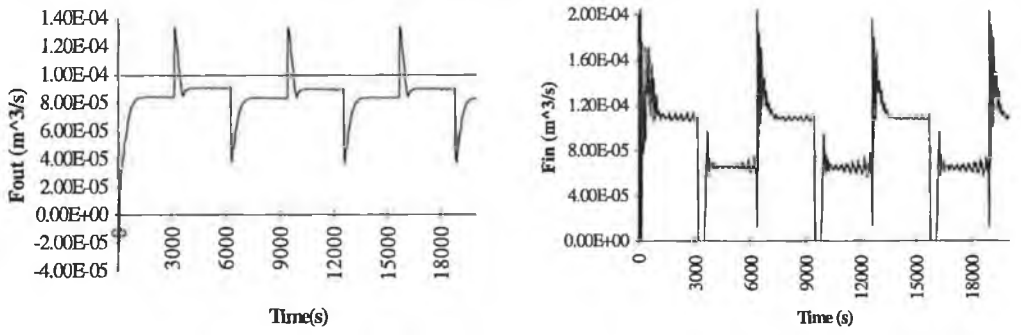


Figure 5.9 - Response of SOC for outlet flow control - SOC Test 5.

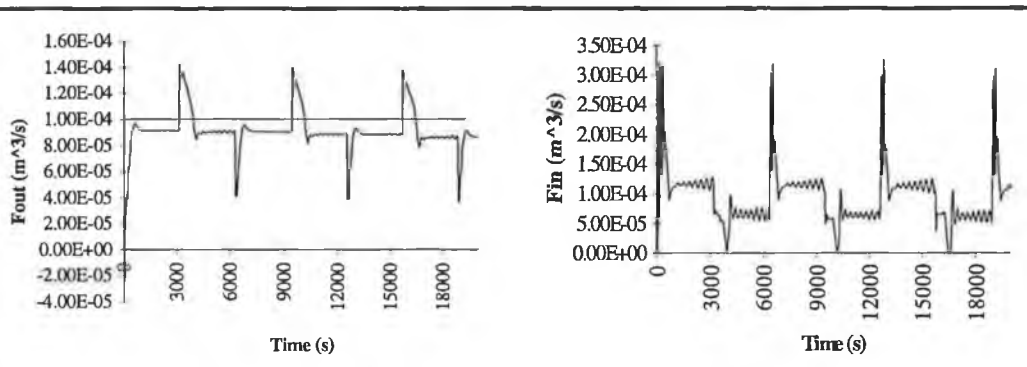


Figure 5.10 - Response of SOC for outlet flow control - SOC Test 6.

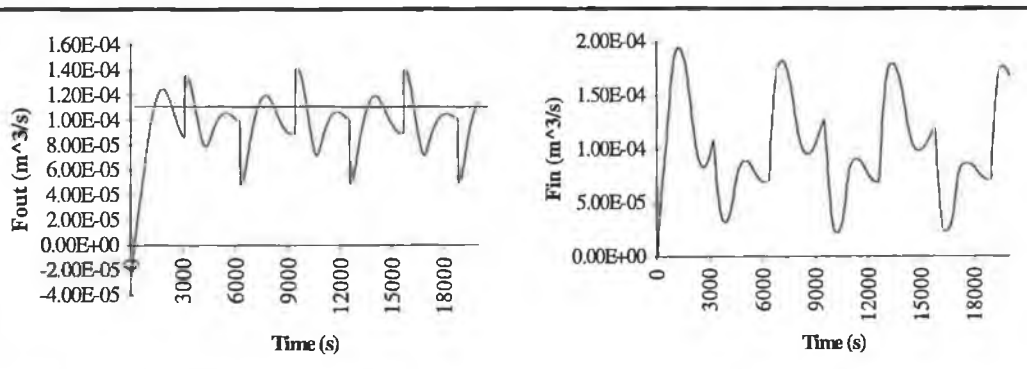


Figure 5.11 - Response of SOC for outlet flow control - SOC Test 7.

Except where otherwise stated, *an empty initial rulebase* was used for all of the SOC simulations. *SOC Test 1* and *SOC Test 2* demonstrate the effect of the value of the learning gain parameter, α, on the quality of control. In SOC Test 1, illustrated in Figure 5.5, where a value of α=5 is used, the speed of learning is fast but the variance in the manipulated variable is high. In SOC Test 2, see Figure 5.6, the value of α has been reduced, α=0.5, with the effect that the speed of learning is decreased and the variance of the manipulated variable is reduced. In *SOC Test 3* the value of α is increased slightly, α=1, the result is improved speed of learning without degradation of the manipulated variable variance, shown in Figure 5.7.

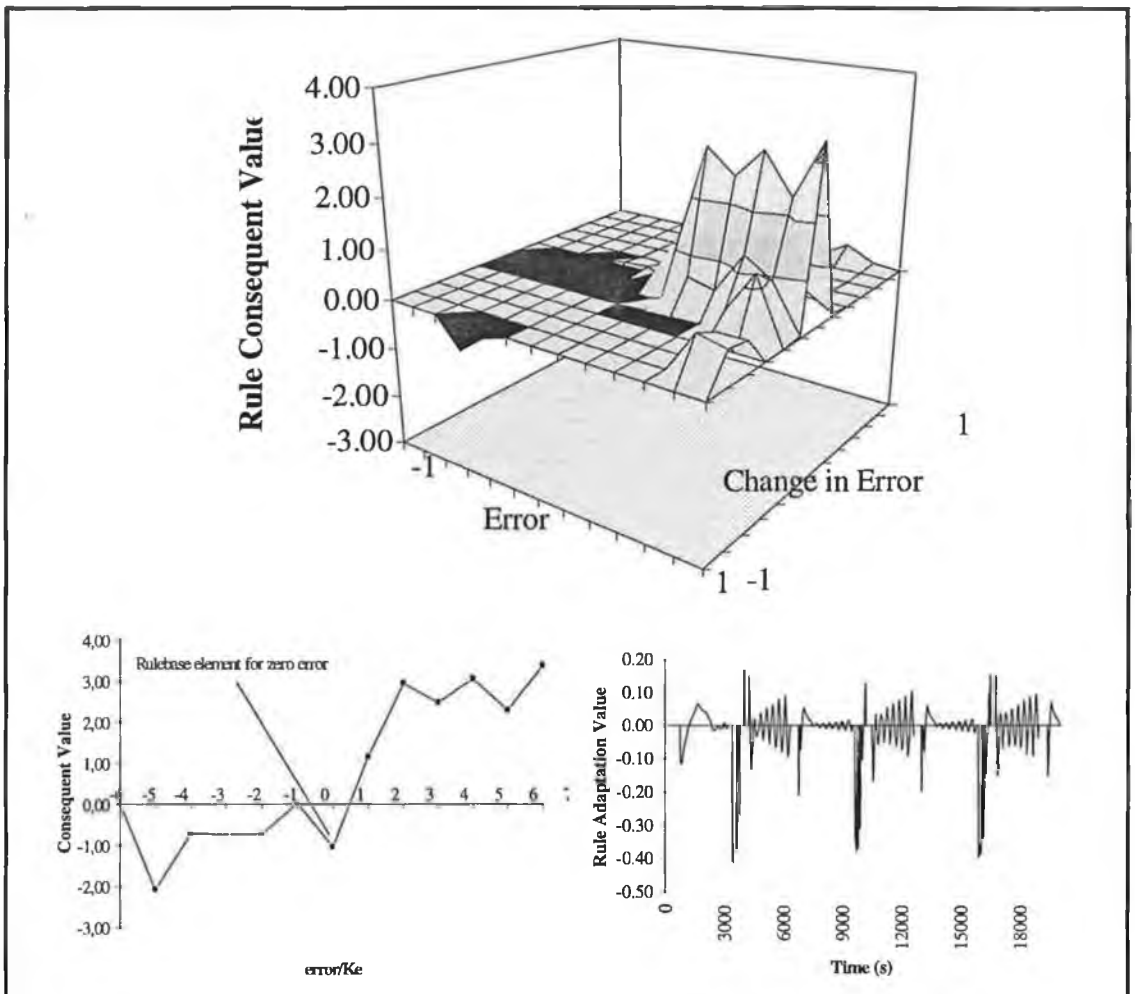


Figure 5.12 - Adapted rulebase, rulebase cross section and rule adaptation values for SOC Test 7.

In order to investigate the effects of initialising the rulebases, *SOC Test 4* utilises the same controller parameters as SOC Test 3, using its adapted rulebase as an initial rulebase. The initial controller performance is better as the initial rulebase is not empty, see Figure 5.8. *SOC Test 5* uses the adapted rulebase of SOC Test 4 as an initial rulebase, whereby no noticeable difference between the results of SOC Test 5 and SOC Test 4 results, as shown in Figure 5.9.

It is deduced from these results that the learning gain, α, influences the speed of learning of the SOC but causes large variance in the controller output if its value is too large. Thus, the value of α must be chosen to give a good compromise between the speed of learning and the controller output variance.

All the previous tests have a steady state offset error in their controller responses. This steady state error was initially attributed to the proportional-derivative nature of the PF-FLC controller used as a basis for the SOC, similar to the classical proportional-derivative controller, which exhibits a steady state error for finite

proportional gains, for type zero systems. In order to attempt to reduce this steady state error, an integral term was introduced into the output gain of the fuzzy controller. *SOC Test 6* shows the results of the best of a set of simulations performed with varying values of the integral and proportional output gains, K_u and K_{ui}, see Figure 5.10. There is some improvement in the steady state error but it still remains at an unacceptable level. *SOC Test 7*, illustrated in Figure 5.11, uses a pure integral output gain, resulting in an overall deterioration of the controller performance.

Based on these results, it was concluded that there is another cause of steady state error in the SOC. After analysis of the rulebase, the steady state error exhibited by all of these simulations was found to be caused by the values of rulebase elements, e.g. the rulebase element which corresponds to zero error has a negative value. This is shown in Figure 5.12, see page 118, where a cross sectional view of the rulebase for a *change in error* value of null is given. Moreover, the value of the rulebase adaptation is seen to oscillate erratically. These characteristics can be attributed to the reference model, which directly specifies the rulebase adaptation.

As the results using this controller all exhibited steady state error, and the influence of the heuristic reference model from Sugiyama [49] on the controller performance is not easily understood, the simple reference model suggested by Spinrad [54] was applied. This reference model is in the form of the product of the controller error and a gain, K_{pi}, thus resulting in a simple and easily understood reference model. After five simulations a value of K_{pi}=2 was found to offer a good compromise between speed of learning and manipulated variable variance. The learning gain and the *error* input variable gain, K_e, are identical to those of SOC Test 6.

After analysis of the adapted rulebases from the previous set of SOC simulations, it was seen that the rulebase information was not sufficiently distributed along the *change in error* rulebase axis, see Figure 5.12. To improve this distribution, the value of the *change in error* gain was increased to K_{de}=200000, thus mapping a maximum *change of error* value of $\pm 0.000005 m^3/s$ to the universe of discourse of [-1,1]. The results from the second iteration of this simulation are shown in Figure 5.13. These results exhibit steady state errors of less than 2% of the setpoint value and are thus far better than any of the previous simulations. This improvement is due to :

- the *change in reference model*, which is one dimensional and directly proportional to the error of the controller and

- *fixation of the centre cell rulebase value at zero*, as suggested by Farbrother, Stacey and Sutton [53].

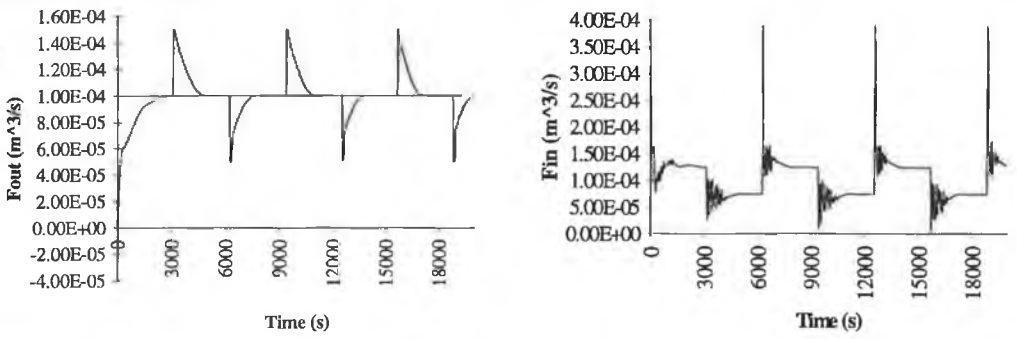


Figure 5.13 - SOC outlet flow control using a scaled error reference model.

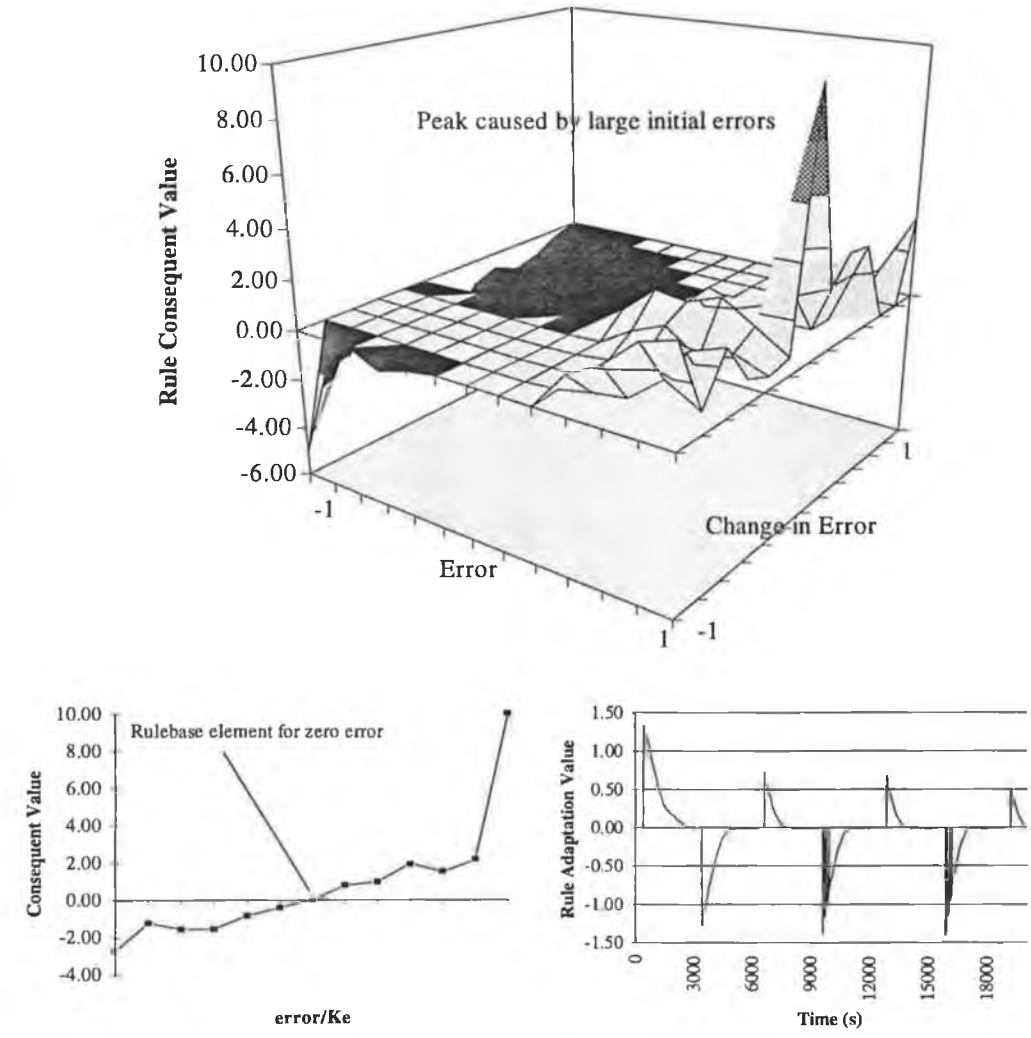


Figure 5.14 - Adapted rulebase, rulebase cross section and rule adaptation values for scaled error reference model.

Figure 5.14, see page 119, shows the rulebase from the SOC using the scaled error reference model as suggested by Spinrad [54]. In addition, a cross section of the rulebase at a *change in error* value of null and a plot of the rule adaptation values are given. The large peak in the rulebase is caused by the adaptation of the cell due to the large errors during the first 3000 seconds. The *cross section of the rulebase* shows a smooth rulebase characteristic with a zero centre value, thus allowing equilibrium at the setpoint and reducing the steady state error. Rule adaptation is directly proportional to the error and is thus devoid of the oscillation seen in the previous simulations. The effect of the increase of the input variable *change in error* gain K_{de} is evident in the distribution of rulebase information the *change in error* axis as well as the *error* axis.

5.2.5. Preliminary Summary of the SOC

The results presented in this section summarise a series of over 150 simulations that were carried out to investigate the capabilities of the SOC controller. The best results obtained exhibited a steady state error of magnitude 2% with a settling time of approximately 1500 seconds. These results alone are reasonable, but when the complexity of the SOC algorithm is considered, and a comparison is made with a standard PI controller, then they are poor. The PI controller gives similar or improved performance without the complexity, large number of parameters and stability problems of the SOC algorithm.

The main disadvantages of the SOC which were encountered during these simulations are :

- *Complexity and memory requirements* of the SOC algorithm. The stored rulebases, for example, require approximately 18kBytes memory.

- *Large number of parameters*, which include the learning gain α, the two input and two output gains, the 78 parameters of the fuzzy membership sets uses for the input variables.

- The *heuristic nature of all of the SOC parameters* can lead to ad-hoc adjustment and long implementation times.

- No *deterministic* method for the specification of the desired response of the control algorithm.

- As far as the author is aware, there is *no method for analysing the stability of the SOC* algorithm although some stability analysis of fuzzy systems has been performed [61].

- The *adaptation of the rulebase is dependent on the position of the controlled variable* and not on its trajectory. Thus if a setpoint change occurs, adaptation is performed although the controlled variable may be moving towards the setpoint in a satisfactory manner. The large spikes in the manipulated variable, see Figure 5.13, for positive error values, are due to the large values in the rulebase for positive error. These large rulebase values, see Figure 5.14, are caused by excessive adaptation of the rulebase during the first 3000 seconds, due to initial outlet flow of zero.

- The reference models found in the literature are of an *ad-hoc nature*, with no thorough analysis having been performed. The scaled error reference model cited by Spinrad and which gave the best results in this chapter exhibits DC rulebase adaptation, even though the controller rulebase uses the change in error as an input.

- The *specification of the causal relationship* between controller response and rule adaptation is heuristic.

At the conclusion of this chapter, Section 5.5, the SOC is compared to the single step predictive fuzzy controller developed in Section 5.3 and an adaptive fuzzy control strategy for the control of the warm water process is chosen.

5.3. Single Step Predictive Fuzzy Control

5.3.1. Introduction

This section describes both the design and simulation of the *Single Step Predictive Fuzzy Controller (SPFC).* This adaptive fuzzy control strategy is a model based controller and is derived mainly from the description of such a method by Moore and Harris [19], see Section 2.8.

The SPFC controller paradigm is developed for multi input single output (MISO) control and multivariable control of the outlet flow and outlet temperature variables of the warm water process respectively. The original design of the controller is enhanced by the following contributions from this research :

- The *reference model* used by Moore and Harris [19] is extended to a linear first order system. This enables the user to more fully define the desired dynamic response of the controlled system.

- The use of the *supervised adaptive fuzzy modelling paradigm* from Section 4.9, thus allowing on-line adaptation of fuzzy model with lookup table rulebases during control.

- The algorithm is applied to a *multivariable control problem.*

In order to investigate the characteristics of the SPFC controller and to evaluate the first two of the three enhancements listed above, the control of the outlet flow of the warm water process is utilised as a testbed. The experience gained through the control of the outlet flow is then applied to the controller for the outlet temperature of the warm water process and finally to the multivariable controller.

5.3.2. SPFC Outlet Flow Control

The SPFC can be viewed a predictive controller which utilises an adaptive fuzzy model of the plant instead of a deterministic model. The main constituents of the SPFC uses as an initial design in this section, shown in Figure 5.15, see page 125, are listed below with explanatory notes :

- *Fuzzy model of the plant and fuzzy model supervisory adaptation algorithm* - comprise an adaptive fuzzy model of the plant to be controlled. In the case of the warm water process, the supervised adaptive fuzzy modelling strategy as described in Section 4.9, has been used to model both the mass flow and the thermal behaviour of the warm water process.

- *Reference model* - In the general case of predictive control, a user defined reference model uses the system setpoint, $\mathbf{Yref}_{n+1}$, and the current value of the controlled variable, **yp**, to calculate the desired plant response over the prediction horizon. When a multi-step prediction horizon is used, then the desired response, $\mathbf{d}_{n+1},\ldots,\mathbf{d}_{ph}$, is a vector with the number of elements equal to the number of steps in the prediction horizon, **ph**. For a first order reference model, these elements of the desired response vector are calculated by the general first order reference model equation (5.1).

for n = 0 to ph - 1

$$d_{n+1} = e^{\frac{T_s}{\tau}} d_n + \left[1 - e^{\frac{T_s}{\tau}}\right] Yref_{n+1} \tag{5.1}$$

where n is the iteration variable,
ph is the length of the prediction horizon,
d_{n+1} is the desired response,
d_n is the past value of the desired response, whereby the initial value, d_n, is set to the current value of the controlled variable, yp,
$Yref_{n+1}$ is the system setpoint,
τ is the desired time constant specified by the user and
T_s is the sampling time of the controller.

For the specific case of the SPFC, with a first order system reference model, a single value of the desired plant output, $\mathbf{d_{n+1}}$, is calculated, as the prediction horizon is a single step, i.e. ph=1. The transfer function of this SPFC specific, first order reference model is given by equation (5.2).

$$d_{n+1} = e^{\frac{T_s}{\tau}} yp + \left[1 - e^{\frac{T_s}{\tau}}\right] Yref_{n+1} \tag{5.2}$$

where yp is the current value of the controlled variable.

Figure 5.15, see page 125, illustrates the function of the first order reference model.

- *Optimisation Algorithm* - is used to calculate the next controller output in such a way that the desired dynamic response is achieved by the system. In the case of the SPFC, this controller block uses the fuzzy model of the plant to calculate the best controller output, $\mathbf{u_{n+1}}$, that minimises the value of the cost function, *Cost(n+1)*. The cost function used is the magnitude of error between the predicted and desired response, equation (5.3).

$$Cost(n+1) = \left|\mathbf{d_{n+1}} - \mathbf{Ym_{n+1}}\right| \tag{5.3}$$

In order to calculate the next controller output, a search is performed over the fuzzy model rulebase by varying the values of the input variable vector, $\mathbf{ui_{n+1}}$, and, using the corresponding fuzzy model output, $\mathbf{Ym_{n+1}}$, evaluating the cost function for the corresponding values of the fuzzy model input vector, $\mathbf{ui_{n+1}}$.

The chosen controller output vector, $\mathbf{u_{n+1}}$, is the fuzzy model input vector, $\mathbf{ui_{n+1}}$, that best minimises the value of the cost function. In the SPFC used in this research, the values of the fuzzy model input vector, $\mathbf{ui_{n+1}}$, are varied by a simple search algorithm. This algorithm first chooses ten equally spaced values for each element of the $\mathbf{ui_{n+1}}$ vector across their respective universes of discourse, evaluates the cost function for each set of fuzzy model input vectors, then performs a fine search around the best three of these vectors. The controller output, $\mathbf{d_{n+1}}$, is the value of $\mathbf{ui_{n+1}}$, that best minimises the cost function value.

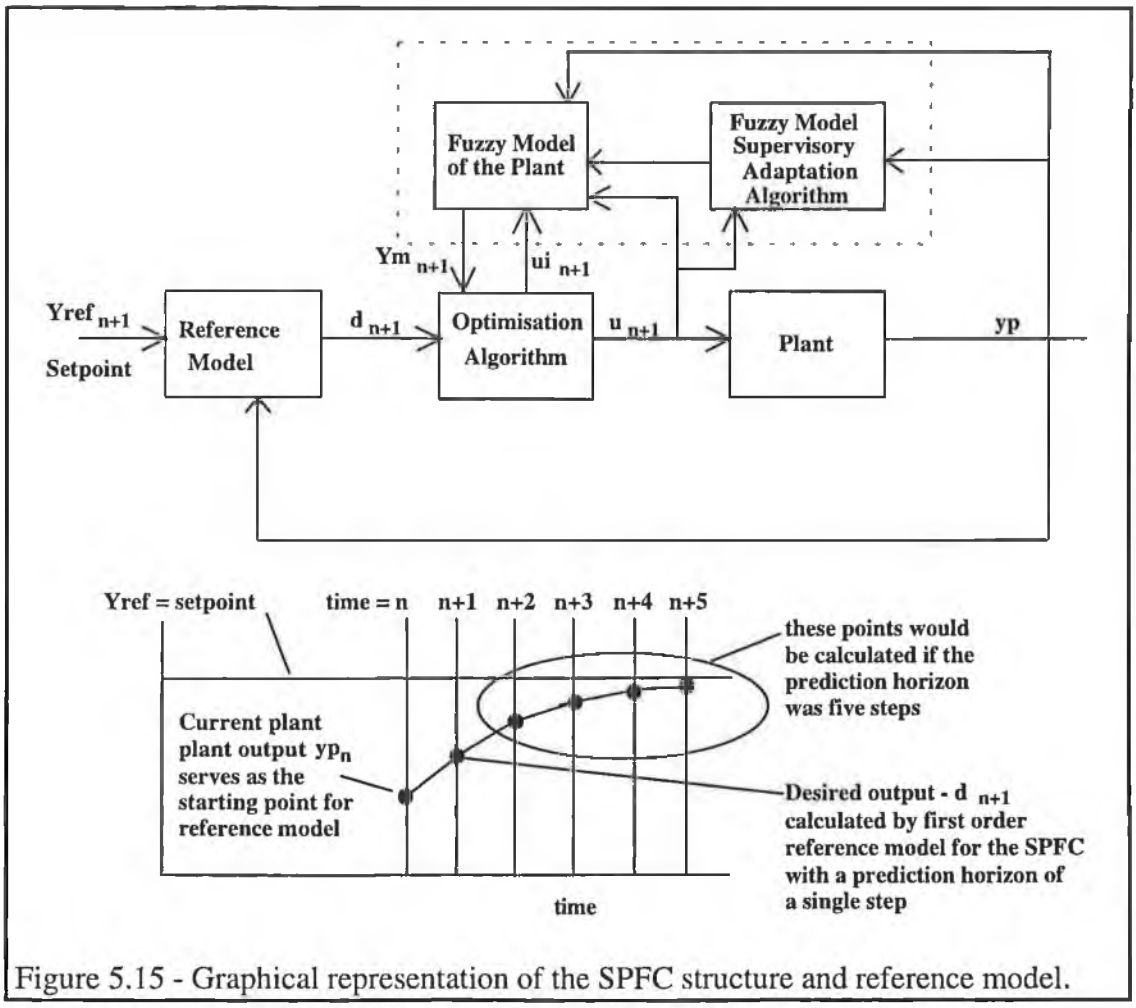


Figure 5.15 - Graphical representation of the SPFC structure and reference model.

All simulation work carried out throughout the SPFC design phase utilised the *ANN model of the warm water process* as a plant, see Section 4.6. The fuzzy models used in the SPFC were initialised using the ANN model of the warm water plant. This initialisation is performed by training the fuzzy models from the ANN model while driving the ANN model with random input values. By means of this training, the fuzzy model learns the behaviour of the ANN model. The reader is referred to Section 4.9.4 of this thesis where this training (fuzzy model initialisation) is described in more detail.

The first controller investigated in this section utilised the *original step reference model* from Moore and Harris [19] and thus attempted to reach the setpoint value of the system within a single controller iteration. The response of this controller with the step reference model is shown is Figure 5.16 for the control of the outlet flow at a setpoint of 100 ml/s.

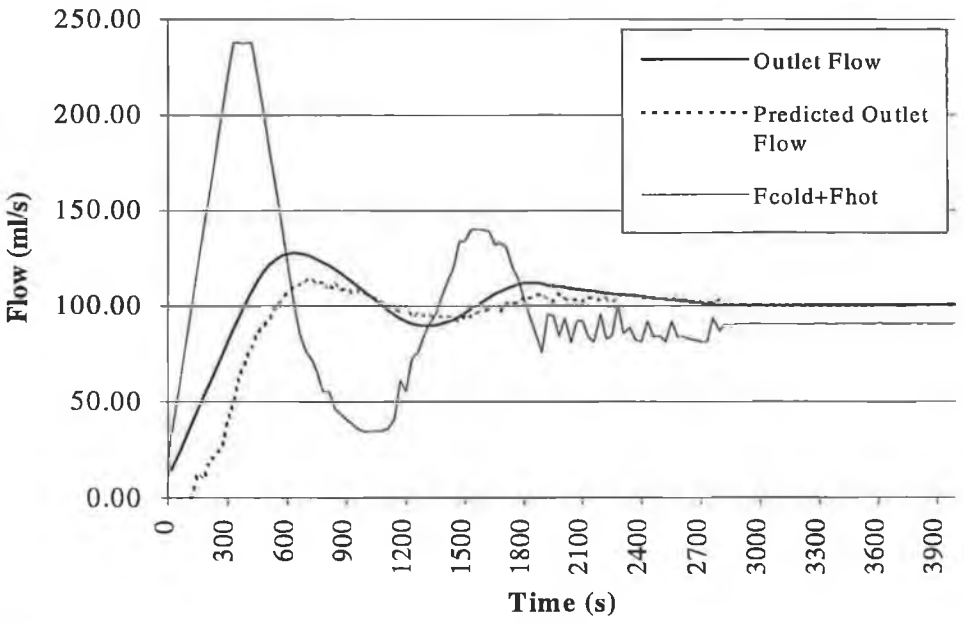


Figure 5.16 - Step reference SPFC outlet flow control with a setpoint of 100 ml/s.

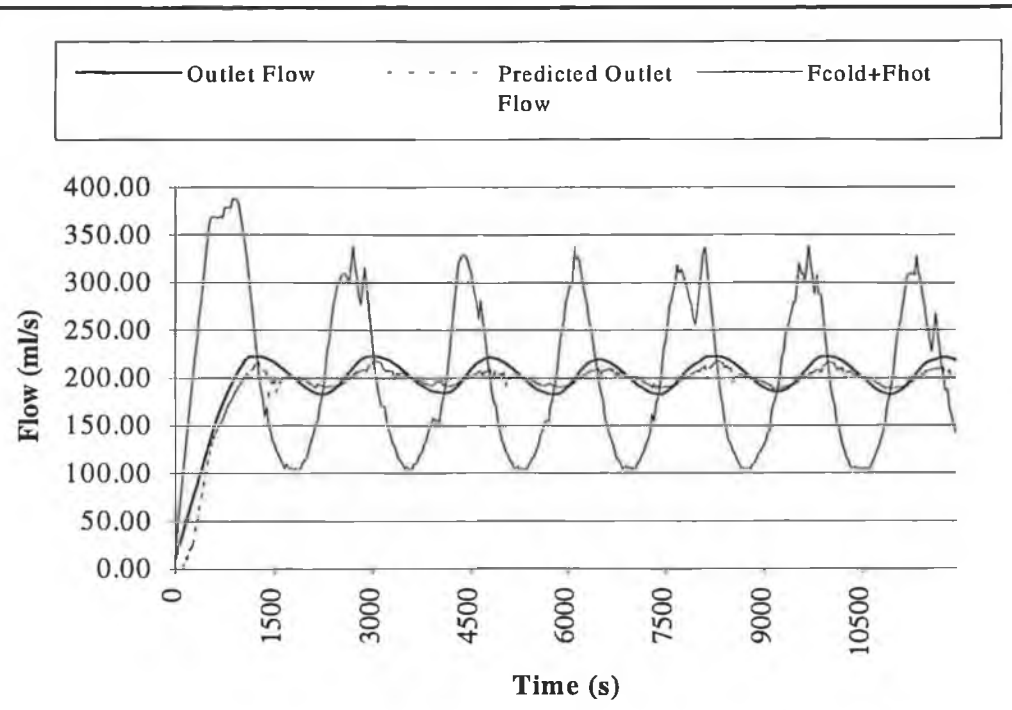


Figure 5.17 - Step reference SPFC outlet flow control with a setpoint of 200 ml/s.

The response of the system shown in Figure 5.16, see page 126, is oscillatory but does settle to the final setpoint value after approximately 2700 seconds. Figure 5.17, see page 126, shows the response of the SPFC controller with a step reference model for a setpoint of 200 ml/s. At this plant operating point, the outlet flow continues to oscillate around the setpoint, even after 10000 seconds.

The oscillation observed in both of these responses is due to the large variations in the manipulated variable, i.e. the inlet flow, which are caused by the controller's utilisation of the step reference model. This can be explained by the fact that the controller with a step reference model has a large forward loop gain.

In order to attempt to reduce the oscillatory nature of these responses, the step reference model from Moore and Harris [19] is replaced with a first order reference model. Through specification of the first order reference model dynamics, more control over the response of the SPFC can be achieved. The first order reference model is defined by equation (5.2), see page 124, where the value of the time constant, τ, is defined by the user. The SPFC with a first order system reference model has a reduced forward loop gain compared to the step reference SPFC, which should thus reduce the oscillation observed with the step reference SPFC.

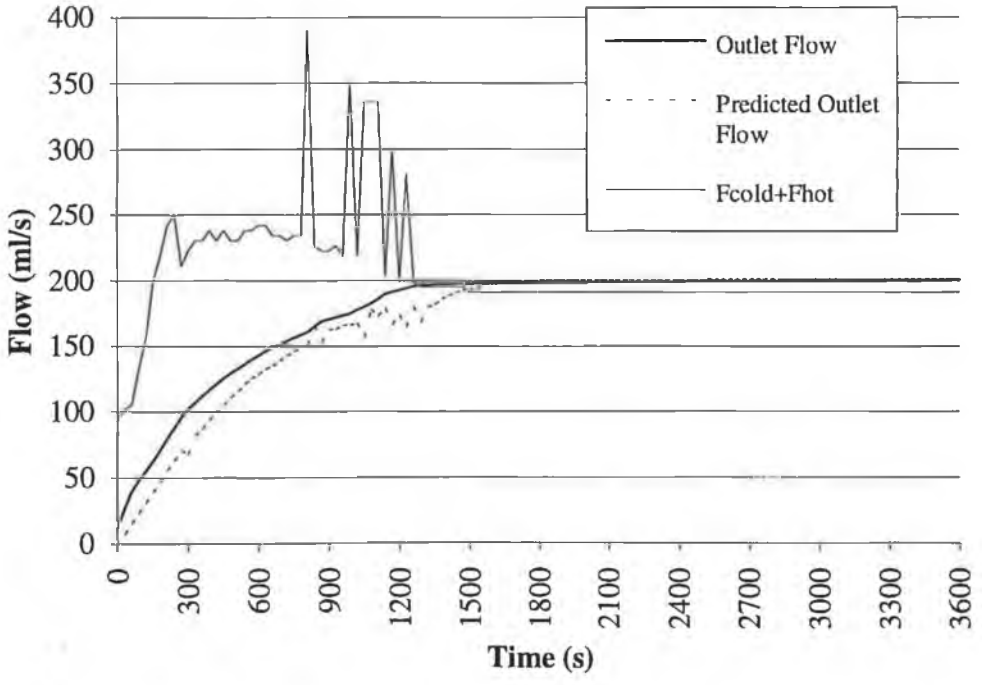


Figure 5.18 - Outlet flow control for 200 ml/s with a first order reference model.

In order to appraise the influence of the first order reference model on the SPFC response, the simulation for outlet flow control with a setpoint of 200 ml/s is repeated. The chosen time constant of the first order reference model for this simulation is τ =600. This time constant value is chosen because it represents the actual plant time constant at an average flow throughput value of 125 ml/s. Figure 5.18, on the previous page, shows the response achieved.

With the first order reference model incorporated into the SPFC design, the response of the controller, for the setpoint of 200 ml/s, no longer exhibits oscillation around the setpoint and settles to a steady state value within 1500 seconds, with a time constant of approximately 480 seconds. Based on this improvement, the first order reference model was incorporated into the controller design.

In an attempt *to increase the accuracy of the fuzzy model* used in the SFPC, and thus improve the SPFC response, the structure of the fuzzy models from Chapter 4 is now modified, and the effect on the response of the SPFC investigated. The new fuzzy model structure predicts the *change in the plant output over one sample* and uses *absolute values* as input variables. This modified fuzzy model structure is represented by the equation (5.4), which uses general variables. The modified fuzzy model structure differs from that of the fuzzy models in Chapter 4, where the absolute value of the modelled variable was predicted and the change in the modelled variable served as a model input, see equation (5.5) and Section 4.9.4.

Modified Fuzzy Model

$$dYm_{n+1} = FuzzyModel[\, u_n, y_n \,] \tag{5.4}$$

where dYm_{n+1} is the predicted change of the modelled system variable, *yp*, over one sample,
u_n represents the plant input variables,
y_n represents the plant output variables and
n is an iteration variable.

Old Fuzzy Model

$$Ym_{n+1} = FuzzyModel[\, u_n, dyp_n \,] \tag{5.5}$$

where Ym_{n+1} is the predicted value of the modelled system variable *yp* and dyp_n represents the change in plant output variables over one sample.

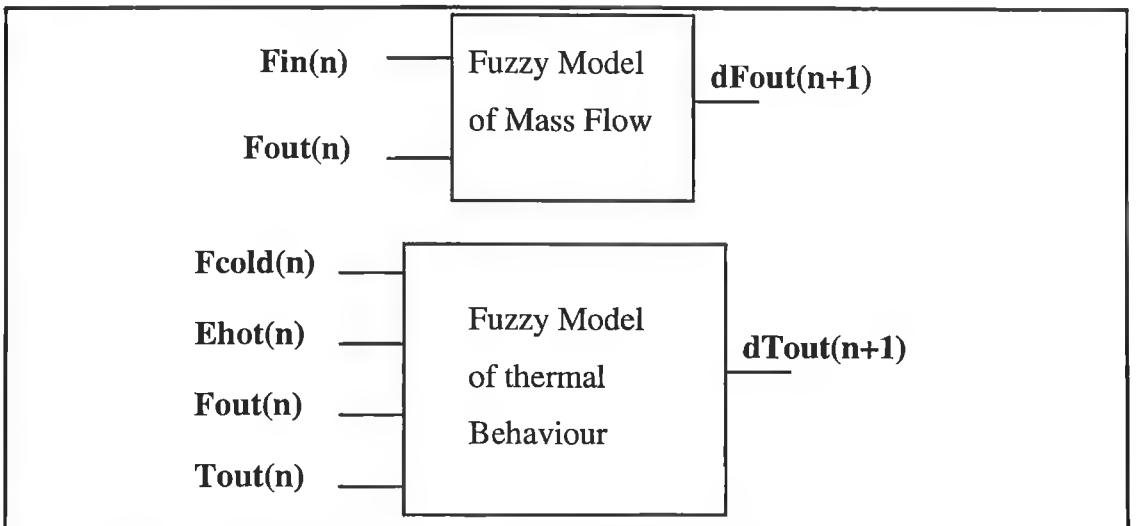


Figure 5.19 - Modified fuzzy models of mass flow and thermal behaviour.

Figure 5.19 shows the *new model structures* for the fuzzy models of the mass flow and thermal behaviour of the warm water process based on equation (5.4). The new mass flow fuzzy model utilises the sum of the hot and cold inlet flows, **Fin(n)**, and the value of the outlet flow, **Fout(n)**, as in inputs to predict the change in the outlet flow over one sample, **dFout(n+1)**. The new fuzzy model of the thermal behaviour of the warm water process uses the absolute values of the hot enthalpy, **Ehot(n)**, the cold flow, **Fcold(n)**, the outlet flow, **Fout(n)**, and the outlet temperature, **Tout(n)**, to predict the change in the outlet temperature over a single sample, **dTout(n+1)**.

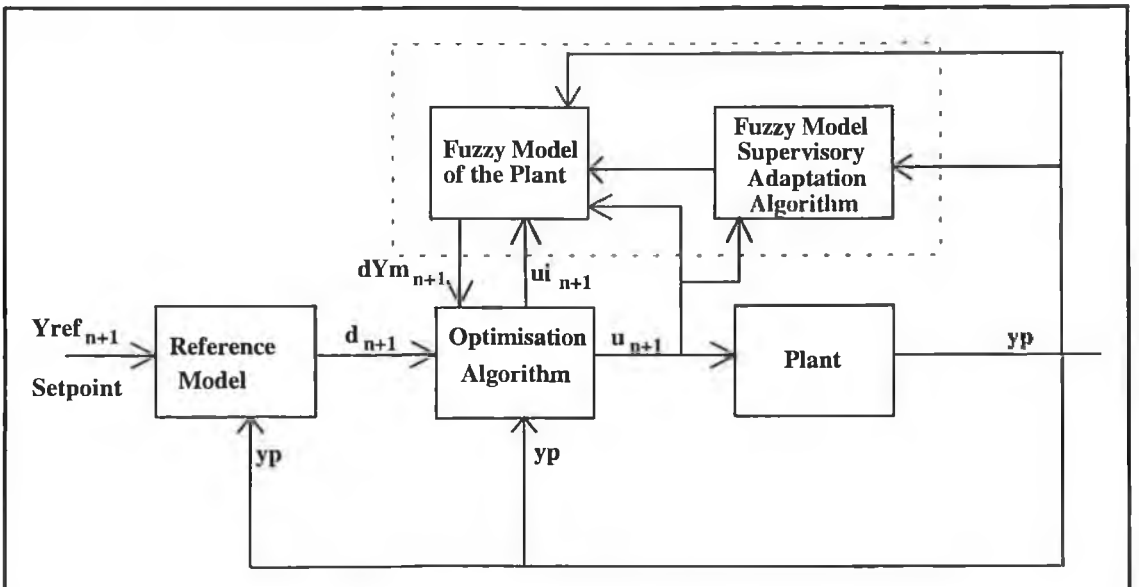


Figure 5.20 - SPFC Structure using the new fuzzy model structure.

In order to use these new fuzzy model architectures, the SPFC structure must be modified. This modification is shown in Figure 5.20, where the real variable is fed to

the optimisation algorithm and used in the cost function. The cost function for this modified SPFC structure is given by equation (5.6).

$$Cost = \left|(\mathbf{dYm}_{n+1} + \mathbf{yp}) - \mathbf{d}_{n+1}\right| \tag{5.6}$$

In order to investigate the effect of the new fuzzy model structure on the SPFC response, the simulated control of the outlet flow using the new SPFC structure for a setpoint of 200 ml/s was carried out. The chosen time constant of the system was again 600 seconds. An improvement in the modelling accuracy is clearly seen when the responses of the old SPFC, Figure 5.18, and the modified SPFC, Figure 5.25, are compared. This modelling improvement is verified by the mean error magnitude of the predicted flow - 4.24 ml/s for the old SPFC and 0.06 ml/s for the modified SPFC. Moreover, the response of the modified SPFC is smoother and, with a time constant of 570 seconds, corresponds more to the desired dynamic response of reference model.

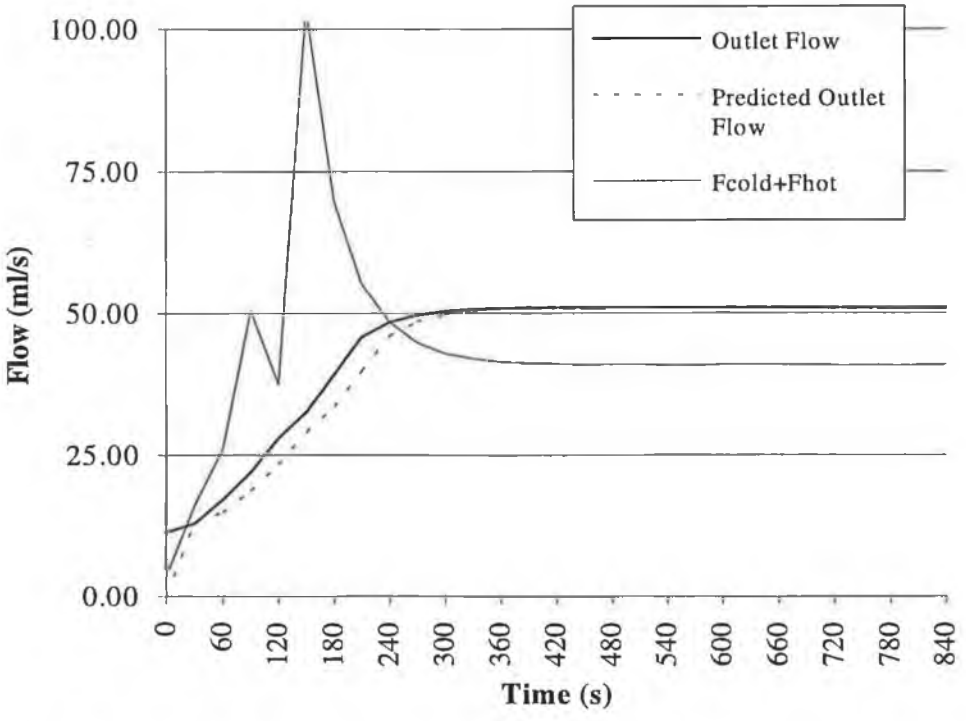


Figure 5.21 - Outlet flow SPFC response for a setpoint of 50 ml/s.

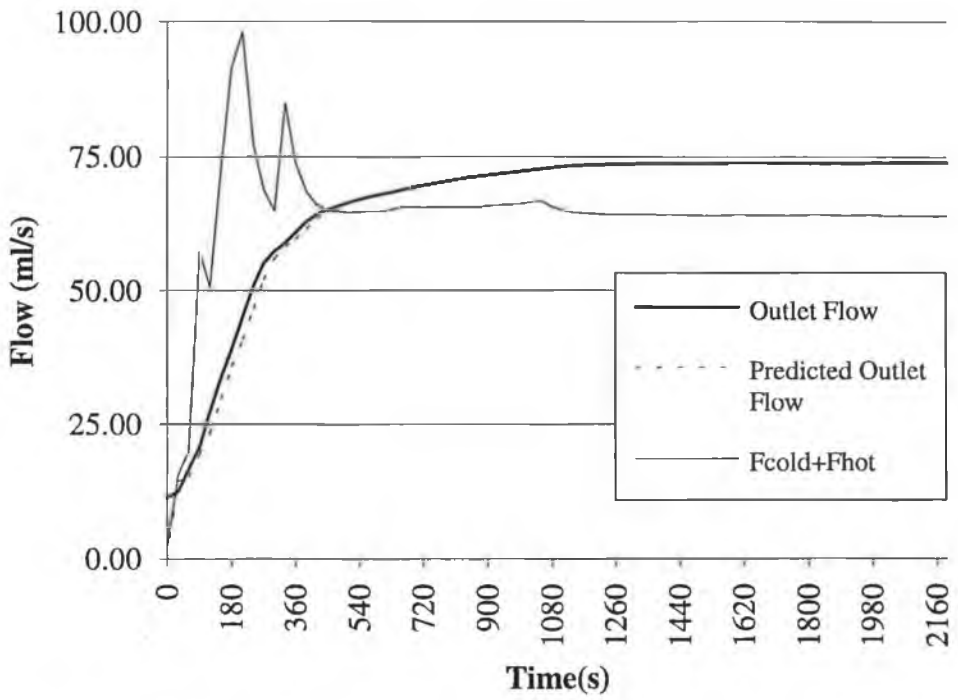


Figure 5.22 - Outlet flow SPFC response for a setpoint of 75 ml/s.

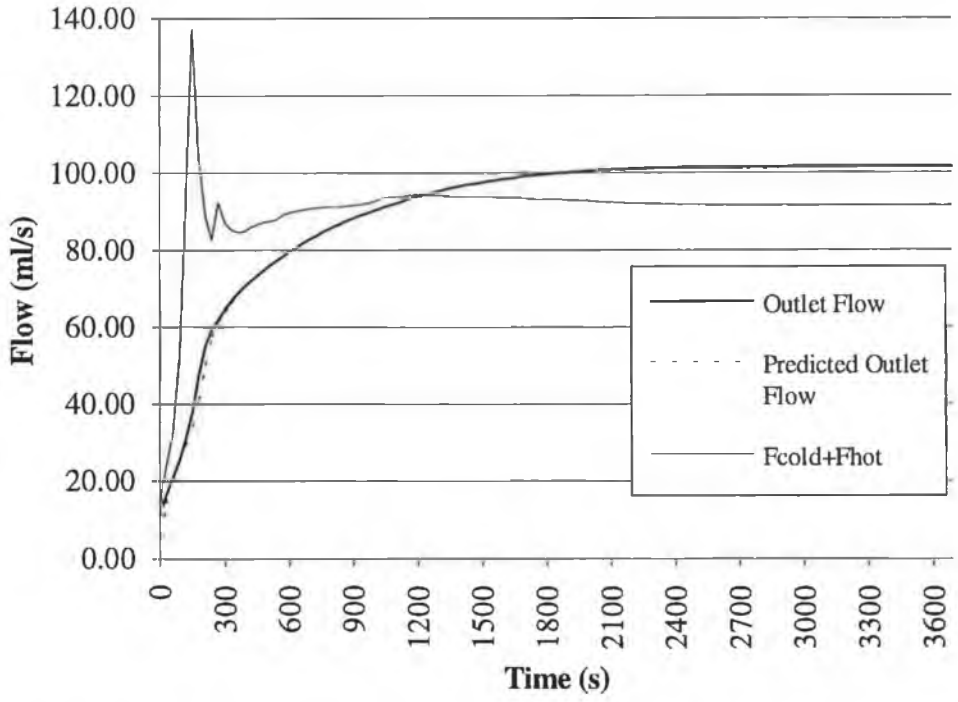


Figure 5.23 - Outlet flow SFPC response for a setpoint of 100 ml/s.

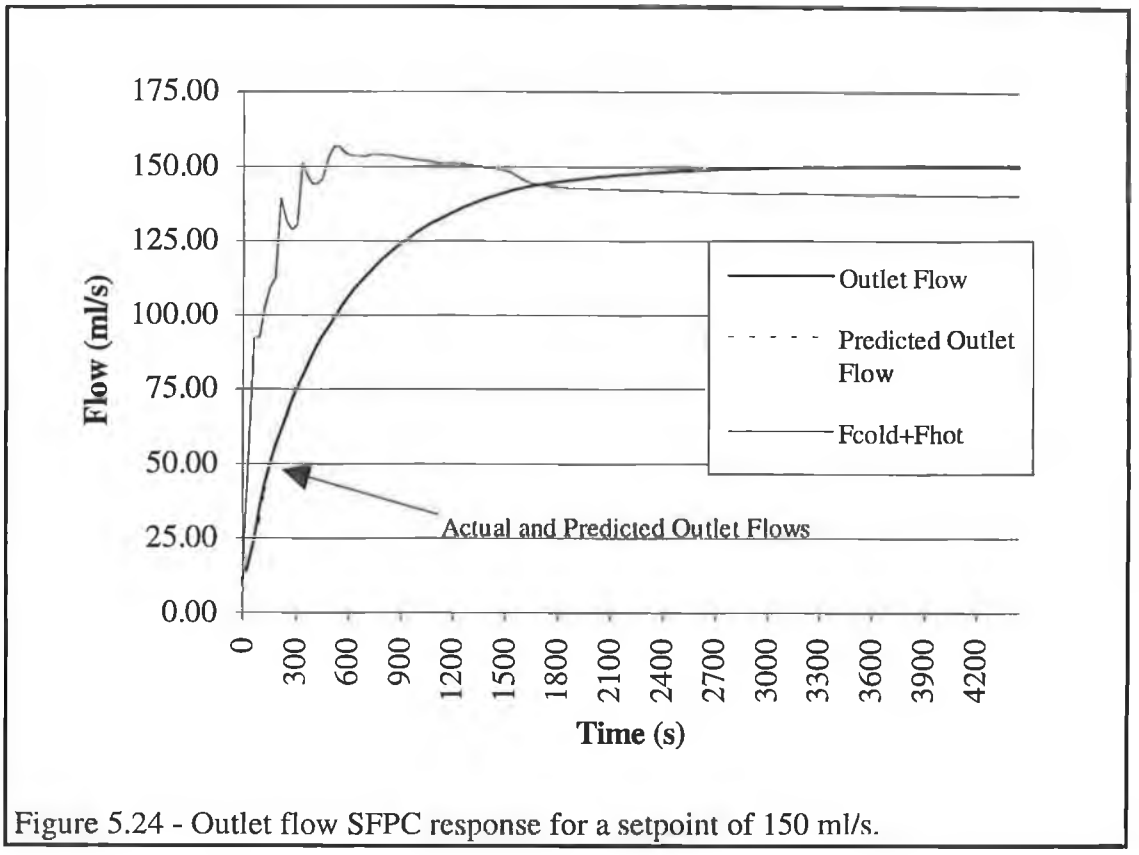


Figure 5.24 - Outlet flow SFPC response for a setpoint of 150 ml/s.

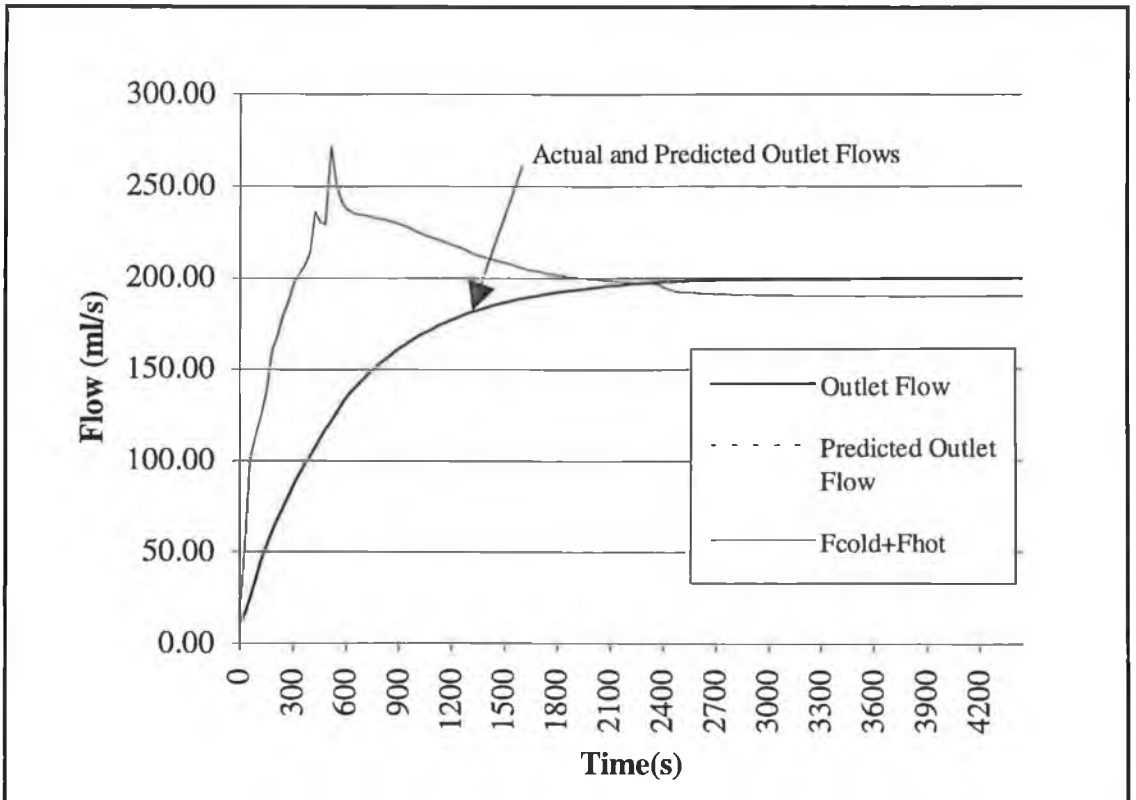


Figure 5.25 - Outlet flow SFPC response for a setpoint 200 ml/s.

Based on the improved fuzzy model accuracy and SPFC response, the fuzzy model formats shown in Figure 5.19, see page 129, and represented by equation (5.4) are to be adopted into the SPFC design. Using this modified SPFC design, the controller responses for the following outlet flow setpoints are simulated : 50 ml/s, 75 ml/s, 100 ml/s and 150 ml/s with an initial outlet flow value of 10 ml/s. The SPFC responses for these four outlet flow setpoints are shown in Figures 5.21 to 5.24. The desired time constant for all of these responses was 600 seconds.

The controller responses shown in Figures 5.21 to 5.25 all exhibit a small steady state error. This is due to the modelling errors of the fuzzy model used for the prediction of the mass flow within the SPFC. Such modelling errors are caused by the following :

- *The fuzziness of the model* - each rulebase cell contains consequent values that represent the modelled variable *within the region* of the state space corresponding to the rulebase cell. The rulebase of the SPFC for outlet flow control has 21x21 cells. Thus each cell can have a consequent value that represents the outlet flow over a range of 4.76% of the universes of discourse of the input variables of the model i.e. 240/21=11.42 ml/s for **Fin(n)** and **Fout(n),** assuming maximum values of 240 ml/s. This *fuzziness* can be reduced by increasing either the number of rulebase cells or the value of the adaptation threshold, δ, in the learning algorithm of the fuzzy model, equation (4.25), which is described in Section 4.9.2.5.

- *Rulebase cell interpolation* - the fuzzy model interpolates between rulebase cells, whereby the type of interpolation depends on the fuzzy algorithm structure i.e. type of fuzzy membership sets, inference function used, defuzzification function. These errors can be reduced by increasing the number of rulebase cells.

In all of the previous SPFC simulations, because the fuzzy model was initially trained using the ANN model of the plant, the fuzzy model corresponded exactly to the simulated plant to be controlled (neglecting the modelling errors described in the previous paragraph). This represents the ideal situation for a model based controller such as the SPFC. In order to investigate the adaptation properties of the SPFC, the physical plant model (see Chapter 4) is used to initialise the fuzzy model instead of the ANN model. This corresponds to initialising the fuzzy model with an erroneous model of the plant to be controlled. The fuzzy models of the plant now differ from the plant to be controlled, as the ANN plant model is used as the simulated warm

water process. If this controller configuration is to provide good control, then *the fuzzy models must adapt to correspond to the ANN model* of the plant.

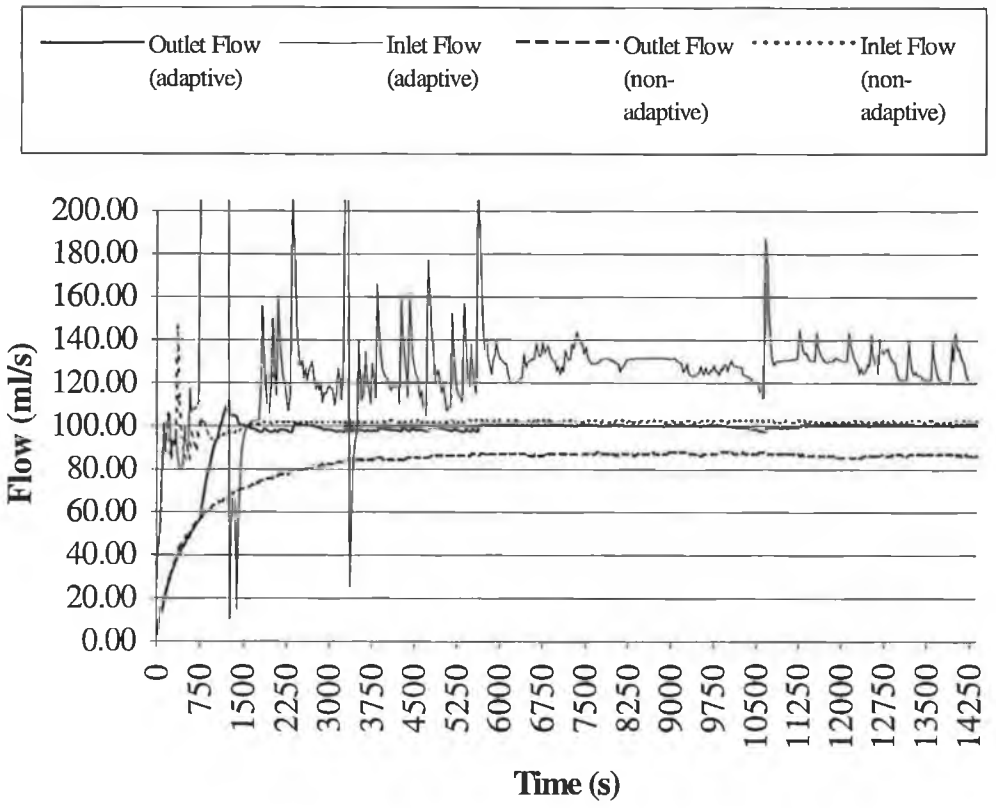


Figure 5.26 - Adaptive and non-adaptive SPFC responses with erroneous fuzzy models.

After initial simulations, the need for a excitation signal to stimulate adaptation of the fuzzy model in the adaptive SPFC became evident. The applied excitation signal is the addition of a white noise to the manipulated variable when a steady state controller error of more than 10% of the setpoint value exists on the plant output.

The effect of this excitation signal on an *adaptive SPFC* is seen clearly in Figure 5.26, where non-adaptive and adaptive SPFCs with erroneous initial fuzzy models are compared. The *non-adaptive* fuzzy controller exhibits a large steady state error, due to the modelling error of the fuzzy model used. Through application of an excitation signal to the manipulated variable, the response of the *adaptive SPFC* converges to the desired setpoint value.

For the *adaptive SPFC*, the added excitation signal must have a large amplitude to allow the adaptation of the fuzzy model around the operating point. The large excitation signal necessary for adaptation of the fuzzy model, causes large fluctuations in the manipulated variable. The large amplitude excitation signal is clearly

disadvantageous when compared to the self-tuning PI controller described in Section 5.4, which due to the utilisation of a deterministic linear plant model, only requires a small amplitude excitation signal.

This section has detailed the development and design of the SPFC using the simulated control of the outlet flow of the warm water process for development and investigation. The design and experience gained through this process are now applied to the simulated control of the outlet temperature of the warm water process in the following Section 5.3.3.

5.3.3. SPFC Outlet Temperature Control

This section presents the results obtained from the *simulated MISO control of the outlet temperature* of the warm water process using the SPFC strategy developed in the previous section. The manipulated variables of the controller are the hot and cold inlet flows of the warm water process. The structure of the fuzzy model used for control of the outlet temperature of the warm water plant is shown in Figure 5.19, on page 129, and is generally represented by equation (5.4), see page 128. This fuzzy model was initialised by training it on the ANN model of the warm water process, which also served as a plant in the simulations described in this section.

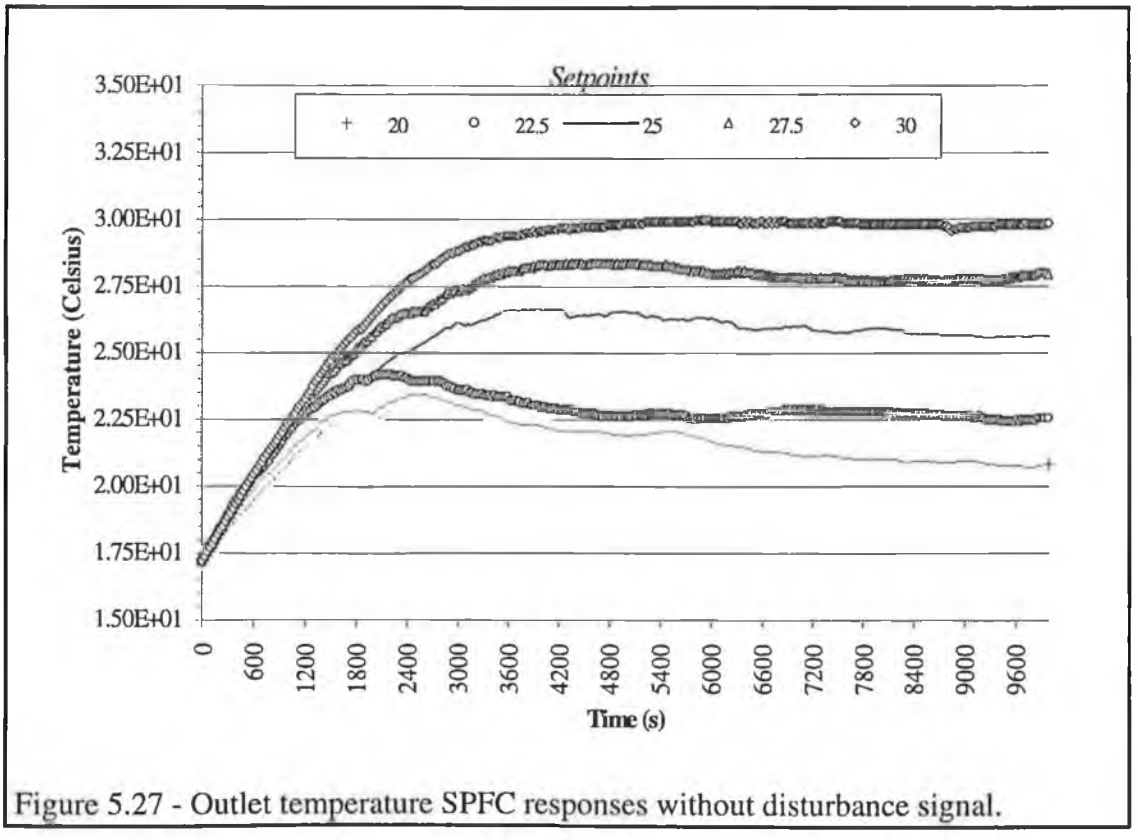


Figure 5.27 - Outlet temperature SPFC responses without disturbance signal.

A set of five simulations for control of the outlet temperature of the warm water process at the following setpoints were carried out : 20, 22.5, 25, 27.5 and 30 degrees Celsius, where the time constant of the first order reference model is 1200 seconds. The results of these five simulations are shown in Figure 5.27.

The controller responses shown in Figure 5.27 are without any disturbance signals on the hot inlet temperature - a constant value of 45 degrees Celsius has been used. Another set of simulations with the same setpoint values were performed but a disturbance signal, shown in Figure 5.28, which was used as the hot inlet temperature variable, which is a measurable plant variable. This *hot inlet temperature disturbance signal* is the sum of three sinusoids and a constant value of 45 degrees Celsius, as shown in equation (5.7). As the cold inlet temperature is not measurable it cannot be used as a disturbance variable.

$$T_{hot} = 45 + 5\sin(2\pi\frac{1}{7000}t + \pi) + 1\sin(2\pi\frac{1}{1000}t) + 0.5\sin(2\pi\frac{1}{300}t) \quad (5.7)$$

where T_{hot} is the hot inlet temperature and
t is the time variable in seconds.

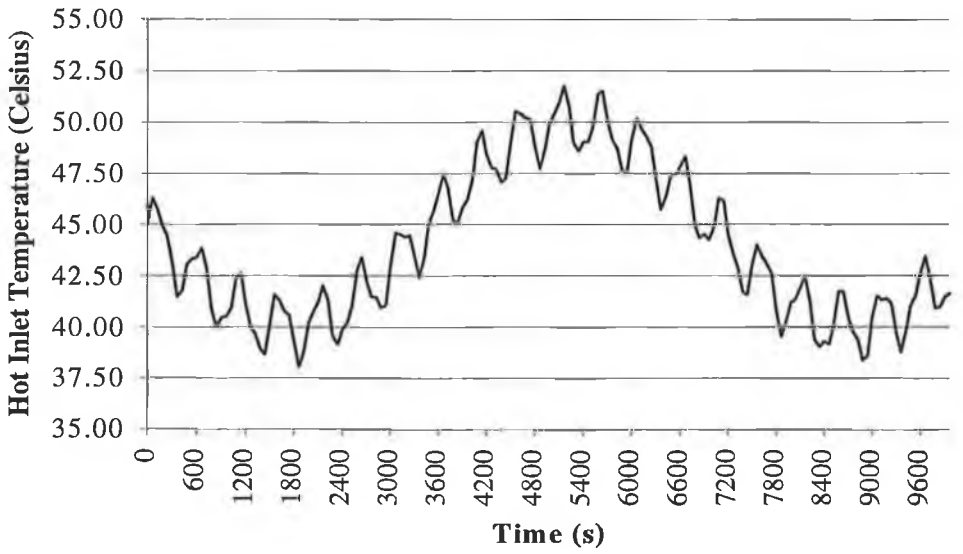


Figure 5.28 - Hot inlet temperature disturbance signal.

This *hot inlet temperature disturbance* signal is based on the empirical observations of the hot inlet temperature variable performed during the course of this research. However, as the temperature of the hot inlet flow is dependent on the hot water use in the rest of the building, this disturbance signal is not a model but only an approximation of possible temperature variations. The new set of controller

responses with this disturbance signal added to the hot inlet temperature is shown in Figure 5.29.

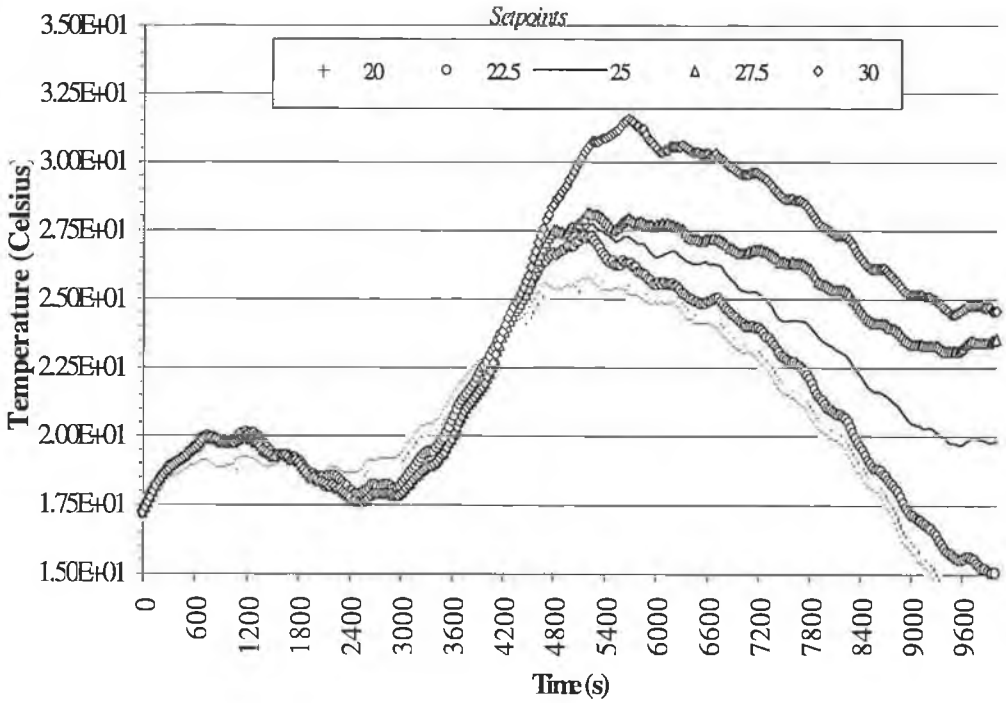


Figure 5.29 - Outlet temperature SPFC responses with disturbance.

The controller characteristics illustrated in Figure 5.29 show that the SPFC for outlet temperature is not robust when a disturbance signal is present on the hot inlet temperature.

The *disturbance rejection* of the controller could be improved if separate measurements of the hot inlet flow and temperature variables were used as fuzzy model input variables. However this would lead to a memory requirement of 3,037,500 bytes of memory (assuming 15 fuzzy membership sets per input variable with 4 bytes per rulebase cell). Unfortunately, this fuzzy model structure is too large for the DOS operating system and cannot be implemented.

Clearly the poor disturbance rejection of the SPFC for outlet temperature control of the warm water process is a serious disadvantage. It remains to be seen how disturbances on the hot inlet temperature on the real warm water process will affect the SPFC response when controlling the outlet temperature of the real plant.

5.3.4. SFPC Multivariable Outlet Flow and Temperature Control

In this section of the chapter, the *multivariable control capability of the SPFC* for the outlet flow and temperature variables of the warm water process is investigated through simulation. The same fuzzy models, as shown in Figure 5.19, and used in the SPFCs for the control of the outlet flow and temperature respectively, are utilised for this multivariable controller.

For both controlled variables, a *separate first order reference model* is used, each with an individual time constant, see equation (5.2), see page 124. The multivariable cost function used for this controller is given by equation (5.8), whereby the quotient terms are used to normalise the flow and temperature terms, which would otherwise have different units and sizes, i.e. m^3/s and degrees Celsius.

$$Cost(n+1) = K\left|\frac{F_{out}^{d}(n+1)-\left(dF_{out}^{m}(n+1)+F_{out}(n)\right)}{F_{out}^{d}(n+1)}\right|+(1-K)\left|\frac{T_{out}^{d}(n+1)-\left(dT_{out}^{m}(n+1)-T_{out}(n)\right)}{T_{out}^{d}(n+1)}\right| \tag{5.8}$$

where n is the iteration variable,
$Cost(n+1)$ is the value to be minimised,
K provides a relative weighting factor for the outlet flow and temperature variables,
$F_{out}^{d}(n+1)$ is the desired response for outlet flow,
$dF_{out}^{m}(n+1)$ is the predicted change in the outlet flow from the fuzzy model for the mass flow of the warm water process,
$F_{out}(n)$ is the real outlet flow,
$T_{out}^{d}(n+1)$ is the desired response for outlet temperature,
$dT_{out}^{m}(n+1)$ is the predicted change in the outlet temperature from the fuzzy model for the thermal behaviour of the warm water process and
$T_{out}(n)$ is the real outlet temperature.

For all simulations described in this section, the *cost function weighting factor K* has a value of 0.5, providing equal weighting for both flow and temperature variables in the cost function.

A set of five simulations for undisturbed multi-variable control of the warm water process using the SPFC were carried out and are described in Table 5.2, see page 145 with corresponding references to the graphs of results. Except where otherwise

stated, a desired time constant of 200 seconds was chosen in order to attempt to speed up the overall plant dynamic response.

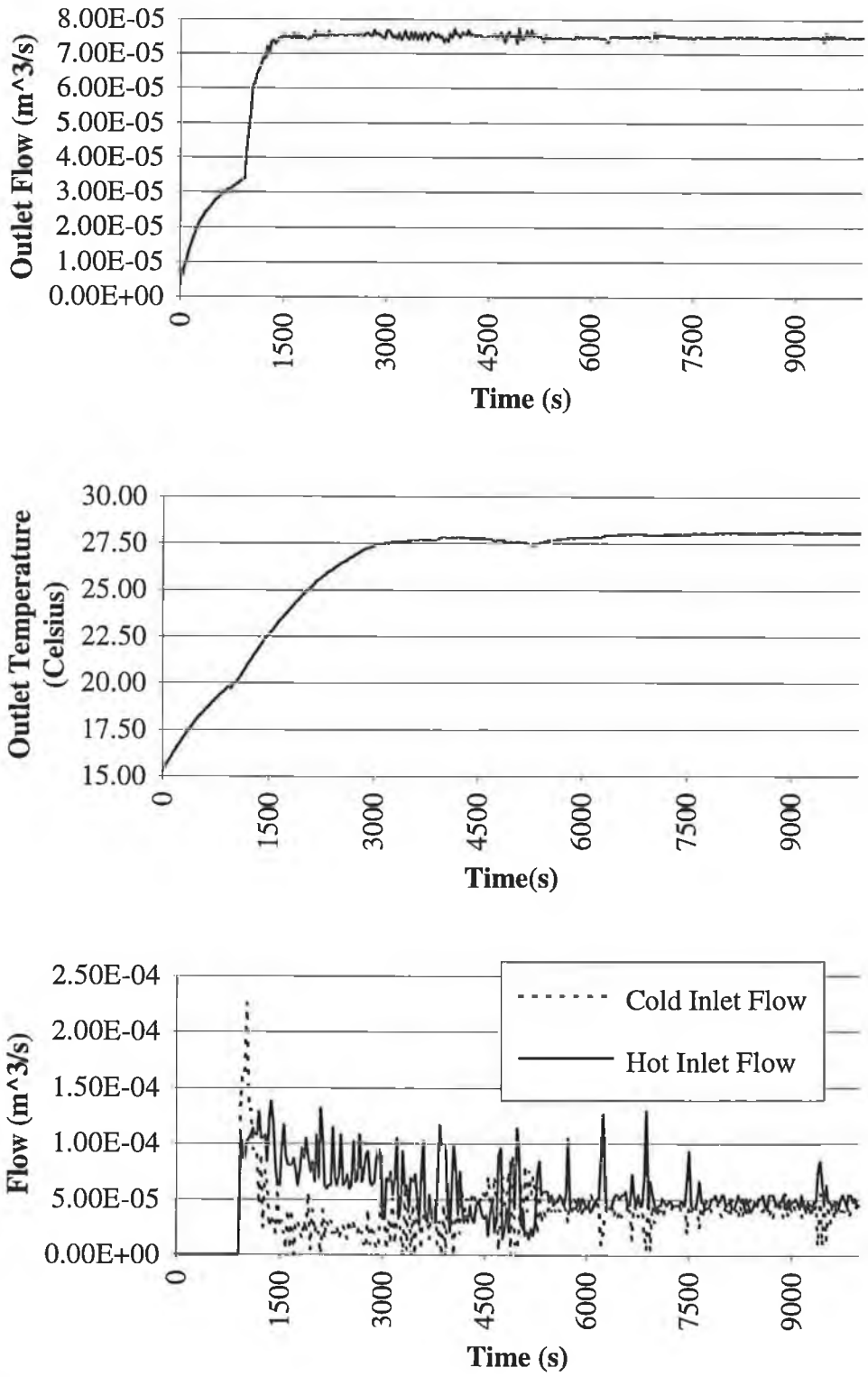


Figure 5.30 - SPFC multivariable control, setpoints = 30 degrees and 75 ml/s.

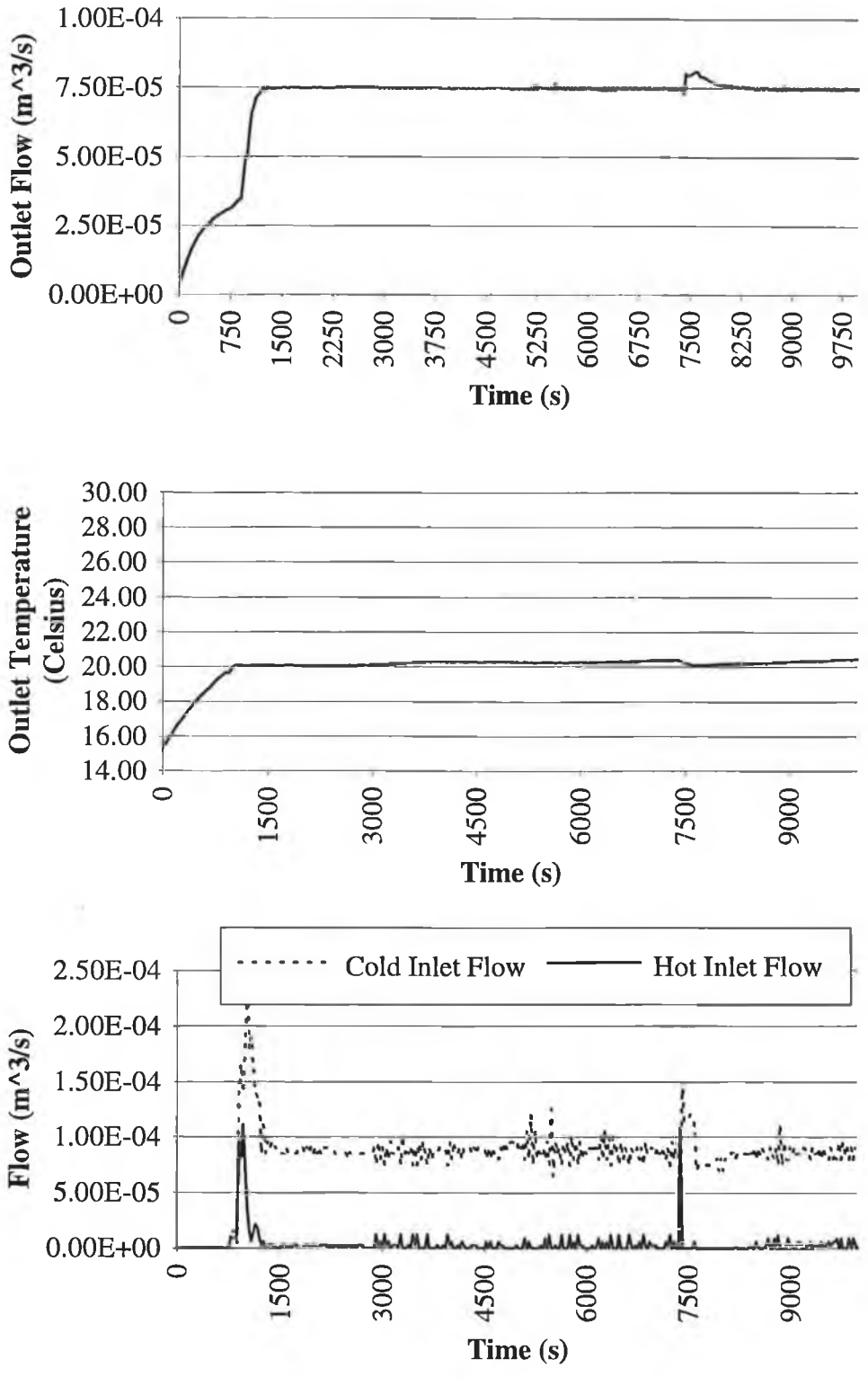


Figure 5.31 - SPFC multivariable control, setpoints = 20 degrees and 75 ml/s.

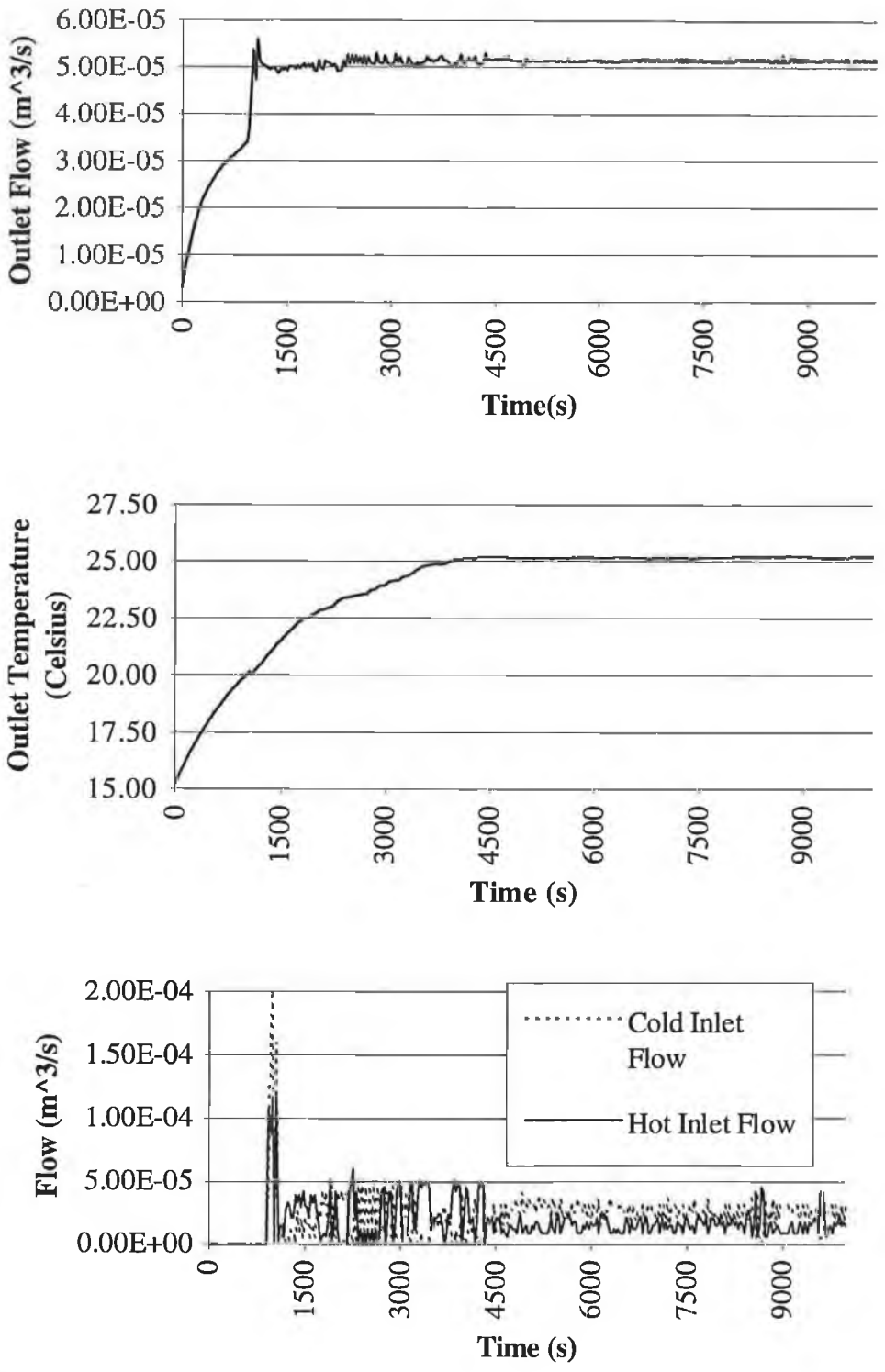


Figure 5.32 - SPFC multivariable control, setpoints = 25 degrees and 50 ml/s.

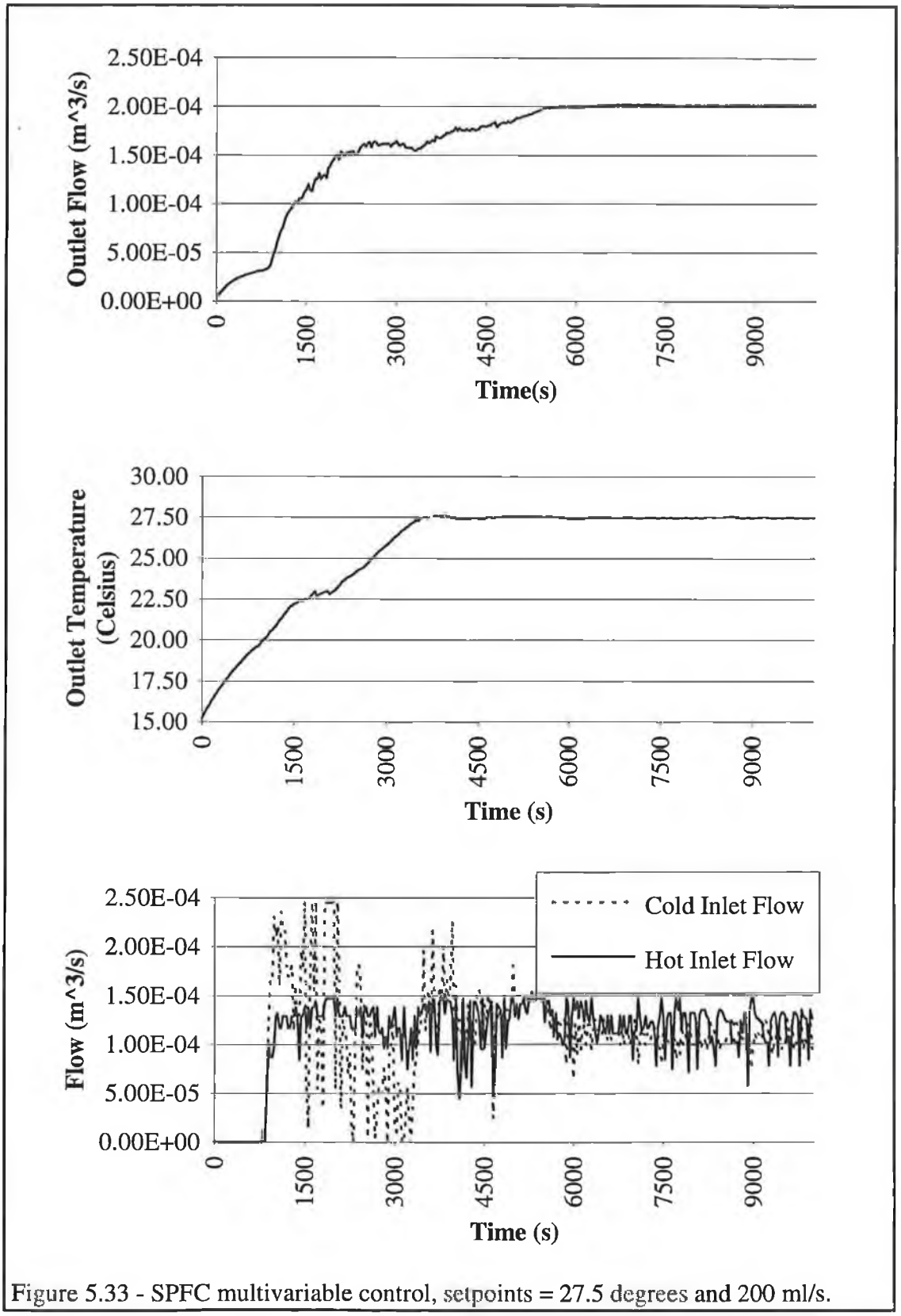


Figure 5.33 - SPFC multivariable control, setpoints = 27.5 degrees and 200 ml/s.

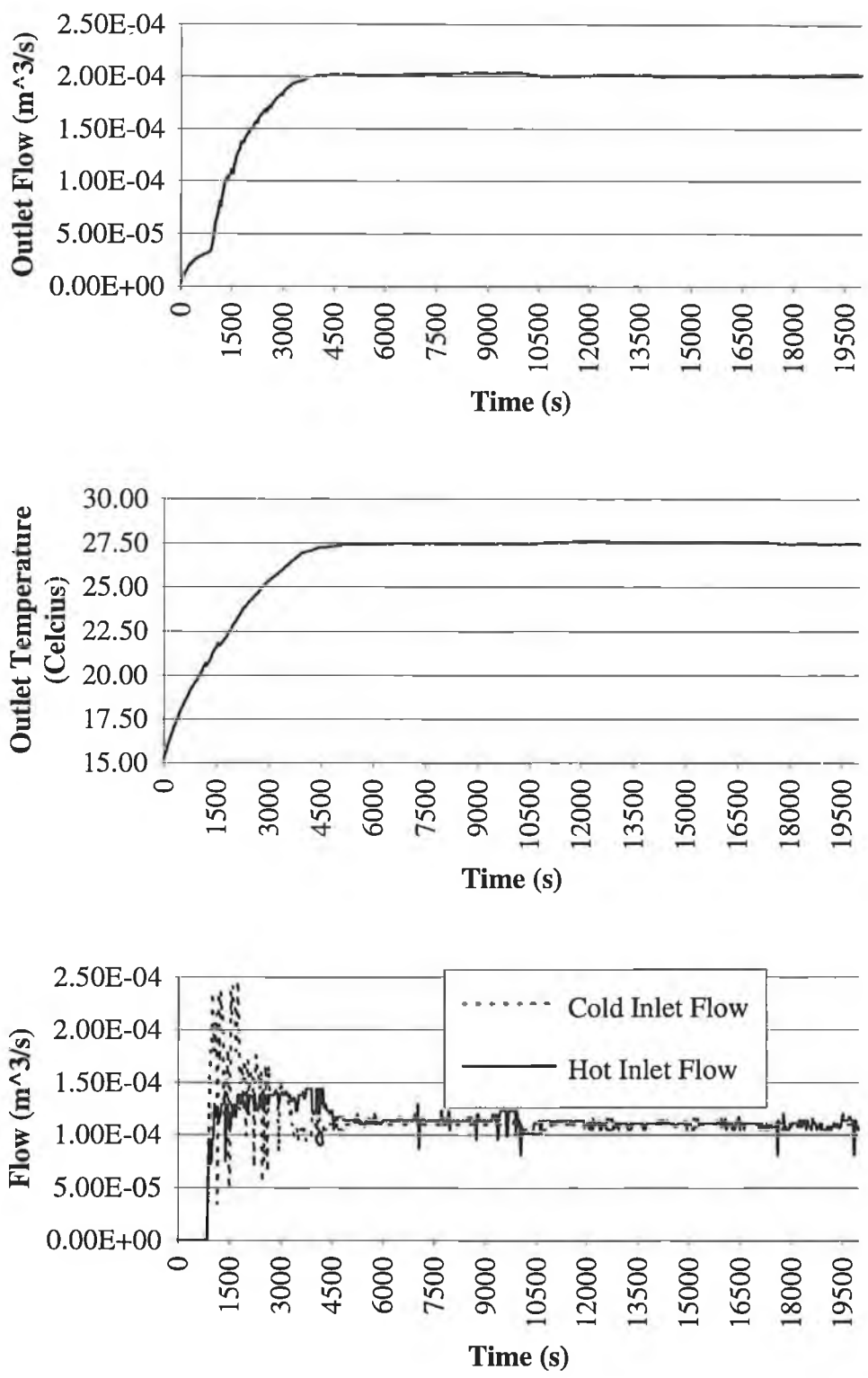


Figure 5.34 - SFPC multivariable control - reference model time constant of 600 seconds.

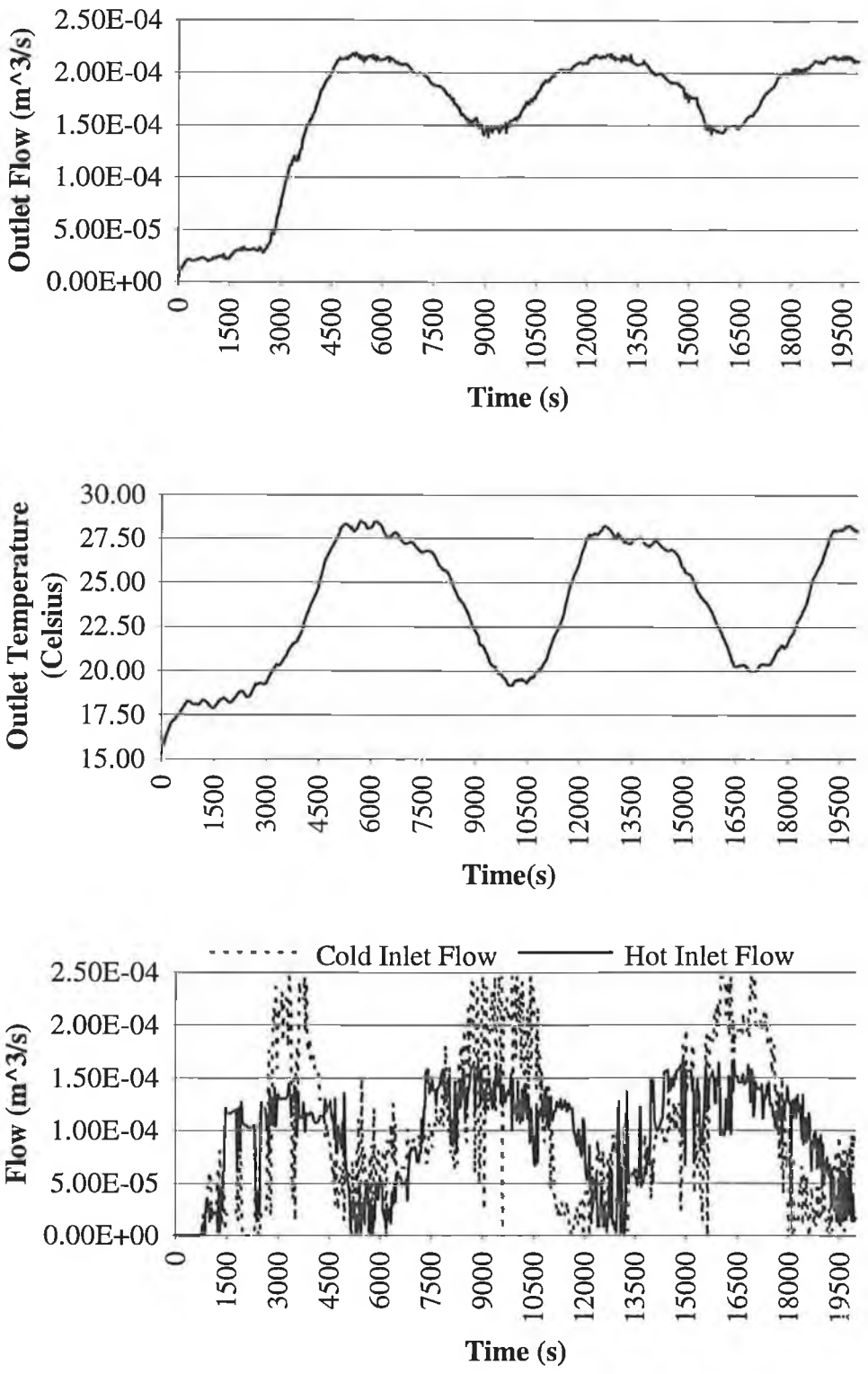


Figure 5.35 - SFPC multivariable control with hot inlet temperature disturbance signal.

Table 5.2 - Multivariable SPFC simulations.

Name	*Outlet Flow Setpoint (ml/s)*	*Outlet Temperature Setpoint (° Celsius)*	*Desired Time Constant (seconds)*	*Graph of Results*
SPFC Test 1	75	30	200	Figure 5.30.
SPFC Test 2	75	20	200	Figure 5.31.
SPFC Test 3	50	25	200	Figure 5.32.
SPFC Test 4	200	27.5	200	Figure 5.33
SPFC Test 5	200	27.5	600	Figure 5.34

There are three types of error evident in the multivariable responses :

- *Steady state error* - caused by the modelling errors of the fuzzy models, the causes of which have been detailed in Section 5.3.1.

- *High levels of variance on the manipulated variables* - caused by the low time constant of the reference model.

- *Disturbances due to coupled nature of the controlled variables* - causing mutual disturbances, which are not compensated by the controller. There are two main causes for this lack of compensation :

    1. The prediction horizon is only a single step, the SPFC cannot predict both the outlet flow and temperature dynamics sufficiently in order to achieve full multivariable control.

    2. The fuzzy model for the thermal response does not contain a full state space representation of thermal behaviour of the plant, only 37% of the rulebase cells have values, see Section 4.9.5.

The effects of the large manipulated variable variance and the coupling of the controlled variables is evident in the response of the SPFC shown in Figure 5.31, see page 140. At approximately 7400 seconds a large cold inlet flow is outputted by the controller in response to the slowly rising temperature. Due to coupling, this large cold input flow then causes a corresponding error in the outlet flow value and a rise in the hot inlet flow value in order to compensate for the (relatively) large downward trend in the outlet temperature caused by the large cold inlet value.

In an attempt to reduce the controller error, SPFC Test 4 was repeated but with a desired time constant of 600 seconds for both controlled variables, (SPFC Test 5) see Figure 5.34. The slower dynamics of the reference model lead to less variance in the manipulated variables, due to the reduced controller gain.

All of the multivariable simulations performed up until now have had no hot inlet temperature variable disturbance, thus representing the ideal plant configuration. To investigate the effect of hot inlet temperature disturbance signals on the multivariable SPFC, the simulated multivariable control of the warm water process with the setpoints of 27.5 degrees Celsius and 200 ml/s was repeated with the hot inlet temperature disturbance shown in Figure 5.28 and with a reference model time constant of 600 seconds for each controlled variable. The results of this simulation are contained in Figure 5.35. As in the case of the single step predictive fuzzy control of the outlet temperature, the multivariable SPFC is not able to control the outlet variables in the presence of the hot inlet temperature disturbance signal. This is due to poor disturbance rejection of the fuzzy model of the thermal behaviour of the warm water process. The coupling evident in the warm water process is seen clearly in the outlet flow response of the SPFC, where large errors are caused by the variance in the manipulated variables due to the controller's attempts to control outlet temperature.

One interesting result of the multivariable simulations, is the fact that the temperature control achieved using the multivariable variable SPFC is of a higher quality that of the MISO control of the outlet temperature. This is due to the fact that the almost constant tank level resulting from the outlet flow control in the multivariable SPFC reduces the non-linear mixing dynamics of the process.

5.3.5. Preliminary Summary of the SPFC

One major disadvantage of the developed SPFC strategy is the search mechanism which requires considerable processing time. If the fuzzy model were differentiable, a gradient descent method could be used. This would allow faster convergence to the optimal controller outputs, thus improving the processing time requirements of the SPFC.

The SPFC strategy would be enhanced if a *Multi-step Predictive Fuzzy Controller* (MPFC) could be developed. As in classical predictive control, the extended prediction horizon of the MPFC would allow improved control of the non-linear warm water process thermal dynamics and offer better multivariable controller performance. Using the current SPFC structure, where a search algorithm is used in

combination with a cost function, more than one step is not feasible due to the exponential increase in time necessary for the search.

Associated with the MPFC, some form of disturbance signal prediction over the prediction horizon would enhance controller performance. For the example of the hot inlet temperature disturbance signal, a prediction of the future hot inlet temperature values could be performed, based on the previous values of the variable.

Further discussion and concluding remarks concerning the SPFC are found in Section 5.6 of this chapter.

5.4. Multi-Step Predictive Fuzzy Control

This section details the development of a strategy for the extension of the single step predictive fuzzy controller (SPFC), developed in Section 5.3, to a multi-step predictive fuzzy controller (MPFC). The motivation for the extension of the prediction horizon is based on the experience gained during simulation in this chapter and in Chapter 6, where the SPFC was unable to adequately control the outlet temperature of the warm water process and did not exhibit the necessary level of compensation for multivariable control of the warm water process. As far as the author is aware, the MPFC introduced in this section is original work.

The extension of the SPFC to form the MPFC offers the advantages of fuzzy model based predictive control with a multi-step prediction horizon. These advantages include :

- *control of systems with non-linear dynamics*, which applies to the thermal behaviour of the warm water process described in this thesis,

- *the facility for multivariable control*, applicable to the multivariable control of the warm water process where the different time scales of the outlet flow and temperature dynamics proved to be the hurdle for multivariable control using the SPFC and

- *look-ahead feature* is useful for systems such as robotics where the future setpoints are known.

In order to realise the MPFC, the following two modifications of the SPFC design are necessary :

- Choice of fuzzy model structure, linguistic functions and parameters to facilitate *differentiation of the fuzzy model.*

- Replacement of the controlled search algorithm, as described in Section 5.3.2, with a *gradient descent algorithm* in order to optimise the controller outputs over the prediction horizon for a given cost function.

5.4.1. MPFC Structure

Figure 5.36 contains a diagram of the suggested MPFC structure with two input variables and a single output variable with a prediction horizon of three samples. The fuzzy model used corresponds to the type introduced in Section 5.3 and represented by equation (5.4) on page 128. This fuzzy model *predicts the change in the modelled variable over one sample.* All variables used in this diagram correspond to those in Figure 5.20 on page 129.

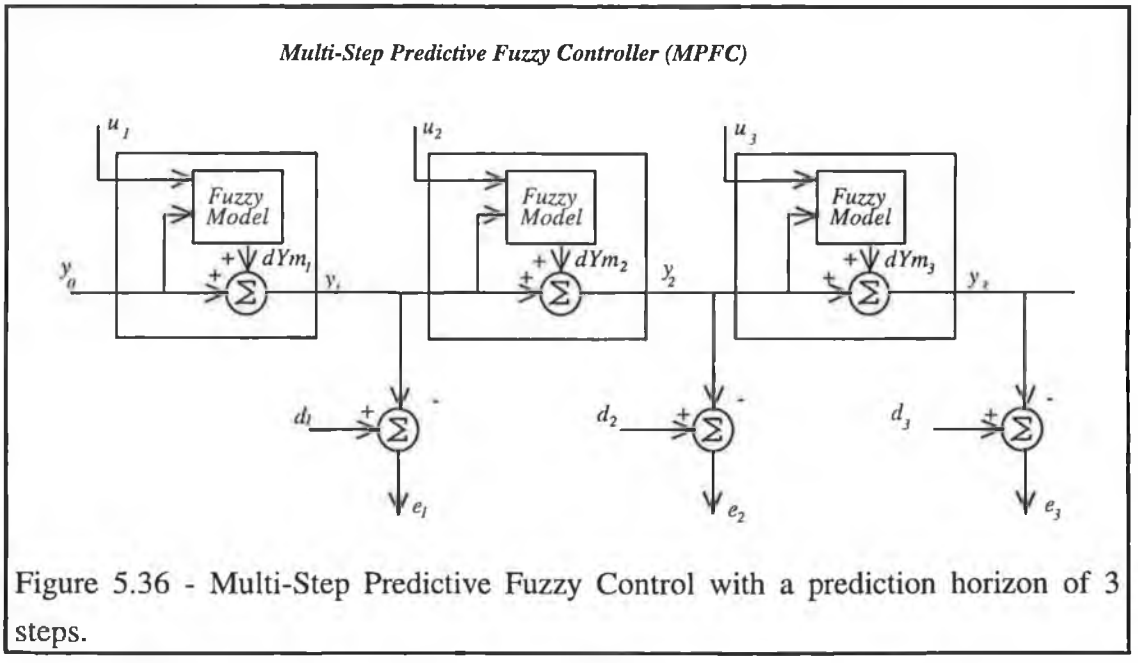


Figure 5.36 - Multi-Step Predictive Fuzzy Control with a prediction horizon of 3 steps.

For the example shown in Figure 5.36, where the fuzzy models are constructed so that their outputs, $\mathbf{dYm_{1..3}}$, are differentiable with respect to their inputs, $\mathbf{u_{1..3}}$ and $\mathbf{y_{0.,2}}$, where $\mathbf{y_0}$ is the current value of the controlled variable, then by applying a gradient descent algorithm to the MPFC structure, it is possible to minimise a given cost function through calculation of the optimal manipulated variable values, $\mathbf{y_{1..3}}$, over the prediction horizon of **ph**, (ph=3 in this example). A simple cost function is given by equation (5.9).

$$E_p = \frac{1}{2}\sum_{n=1}^{ph} \beta_n e_n^2 \tag{5.9}$$

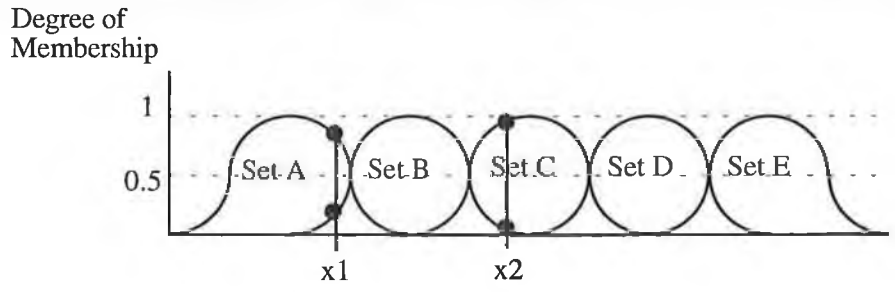
where n is an iteration counter, n=1 ... ph, where ph is the prediction horizon,

e_n is the nth error equal to the difference between the corresponding reference signal, $\mathbf{d_n}$, and the predicted output of the fuzzy model, $\mathbf{y_n}$, and

β_n is the nth weighting factor.

5.4.2. Fuzzy Model Structure

The structure of a fuzzy model suitable for differentiation is described in this section. The chosen model has two independent input variables, u and x, and a single output variable, y. The fuzzy model utilises Gaussian antecedent fuzzy membership sets, whereby *each input variable only activates two fuzzy membership sets*, with all other fuzzy membership sets on the corresponding universe of discourse inactive. Gaussian functions are used as they differentiable. The activation of only two fuzzy sets on the universe of discourse is achieved by using a regularly spaced distribution of fuzzy membership sets and is illustrated in Figure 5.37. It should be noted, that a fuzzy model with any number of inputs and fuzzy membership sets can be used, the example fuzzy model given here has been kept simple, in order to limit the derivation complexity.



The two crisp input variables, x1 and x2, both activate only two of the five fuzzy membership sets. x1 activates Set A and Set B while x2 activates Set B and Set C. This is achieved through a regularly spaced. distribution of fuzzy membership sets, with all inter set intersections at a degree of membership value of 0.5.

Figure 5.37 - Regularly space fuzzy membership set distribution.

A lookup table rulebase format with fuzzy singleton consequent values is utilised. The corresponding degrees of membership for each input variable are given by (5.10), (5.11), (5.12) and (5.13).

$$\mu_1^x = \exp\left(-\left(\frac{x - c_1^x}{\sigma_1^x}\right)^2\right) \tag{5.10}$$

$$\mu_2^x = \exp\left(-\left(\frac{x - c_2^x}{\sigma_2^x}\right)^2\right) \tag{5.11}$$

$$\mu_1^u = \exp\left(-\left(\frac{u - c_1^u}{\sigma_1^u}\right)^2\right) \tag{5.12}$$

$$\mu_2^u = \exp\left(-\left(\frac{u - c_2^u}{\sigma_2^u}\right)^2\right) \tag{5.13}$$

where μ_q^p is the qth degree of membership of the variable p and c and σ are the centre and standard deviation values of the Gaussian fuzzy membership set functions.

These four degrees of membership activate four rules of all possible rules in the fuzzy model. Each of these four rules has an activation level which is calculated by use of an inference function. To give a differentiable function, the *product* operator is used as the inference function for the linguistic function AND. The activation levels of the four rules are given by the equation set (5.14).

$$\begin{aligned} \mu_{11} &= \mu_1^u \mu_1^x \\ \mu_{12} &= \mu_1^u \mu_2^x \\ \mu_{21} &= \mu_2^u \mu_1^x \\ \mu_{22} &= \mu_2^u \mu_2^x \end{aligned} \tag{5.14}$$

Each of the four rules has a corresponding consequent value S, which is contained within individual cells of the rulebase. The output of the fuzzy model, y, is given by equation (5.15) when the centre of gravity defuzzification method for fuzzy singletons is applied.

$$y = \frac{\mu_{11}S_{11} + \mu_{12}S_{12} + \mu_{21}S_{21} + \mu_{22}S_{22}}{\mu_{11} + \mu_{12} + \mu_{21} + \mu_{22}} \tag{5.15}$$

The first derivative of the fuzzy model output, y, with respect to the input variable, u, is required. Through substitution of the equations (5.14) into the equation (5.15), equation (5.16) is arrived at.

$$y = \frac{\mu_1^u \mu_1^x S_{11} + \mu_1^u \mu_2^x S_{12} + \mu_2^u \mu_1^x S_{21} + \mu_2^u \mu_2^x S_{22}}{\mu_1^u \mu_1^x + \mu_1^u \mu_2^x + \mu_2^u \mu_1^x + \mu_2^u \mu_2^x} \tag{5.16}$$

Rearranging for functions of u and substituting the constants K_1^x, $K_2^x$and K_3^x given by equations (5.17),(5.18) and (5.19) into equation (5.16) results in the equation (5.20).

$$K_1^x = \mu_1^x S_{11} + \mu_2^x S_{12} \tag{5.17}$$

$$K_2^x = \mu_1^x S_{21} + \mu_2^x S_{22} \tag{5.18}$$

$$K_3^x = \mu_1^x + \mu_2^x \tag{5.19}$$

$$y = \frac{K_1^x \mu_1^u + K_2^x \mu_2^u}{K_3^x \left(\mu_1^u + \mu_2^u\right)} \tag{5.20}$$

5.4.3. Fuzzy Model Gradient

The first derivative of equation (5.20) with respect to the input variable u is found by utilising the quotient rule for differentiation. For notational purposes, let $\mu_n^{u'} = \frac{d\left(\mu_n^u\right)}{du}$, which is the first derivative of the membership value of a Gaussian fuzzy membership set with respect to the input variable u. Applying the quotient rule to equation (5.20) gives :

$$\frac{dw}{du} = K_1^x \mu_1^{u'} + K_2^x \mu_2^{u'}$$

$$\frac{dv}{du} = K_3^x \left(\mu_1^{u'} + \mu_2^{u'}\right)$$

$$\frac{dy}{du} = \frac{K_3^x \left(\mu_1^u + \mu_2^u\right)\left(K_1^x \mu_1^{u'} + K_2^x \mu_2^{u'}\right) - \left(K_1^x \mu_1^u + K_2^x \mu_2^u\right)\left(K_3^x \left(\mu_1^{u'} + \mu_2^{u'}\right)\right)}{K_3^{x2}\left(\mu_1^{u2} + 2\mu_1^u \mu_2^u + \mu_2^{u2}\right)}$$

$$\frac{dy}{du} = \frac{\left(\mu_1^u + \mu_2^u\right)\left(K_1^x \mu_1^{u'} + K_2^x \mu_2^{u'}\right) - \left(K_1^x \mu_1^u + K_2^x \mu_2^u\right)\left(\mu_1^{u'} + \mu_2^{u'}\right)}{K_3^x \left(\mu_1^u + \mu_2^u\right)^2} \tag{5.21}$$

The equation for first derivatives of the Gaussian fuzzy membership functions with respect to the input variables, $\mu_n^{u'}$ and $\mu_n^{x'}$, now need to be found. These are derived through the application of the product rule for differentiation of an exponential function.

From (5.12),

$$\mu_n^{u'} = \frac{1}{\sigma_n^{u2}}\left(2c_n^u - 2u\right)\exp\left(-\left(\frac{u-c_n^u}{\sigma_n^u}\right)^2\right) \tag{5.22}$$

$$\mu_n^{u'} = \frac{2}{\sigma_n^{u2}}\left(c_n^u - u\right)\mu_n^u \tag{5.23}$$

Similarly $\mu_n^{x'}$ may be found as:

$$\mu_n^{x'} = \frac{2}{\sigma_n^{x2}}\left(c_n^x - x\right)\mu_n^x \tag{5.24}$$

For the derivative of the fuzzy model with respect to x we define K_1^u, K_2^u and K_3^u as in (5.25), (5.26) and (5.27). Rearranging (5.16) for functions of x and after substitution of (5.25), (5.26) and (5.27), equation (5.28) results, which is the first derivative of the fuzzy model with respect to the input variable x.

$$K_1^u = \mu_1^u S_{11} + \mu_2^u S_{12} \tag{5.25}$$

$$K_2^u = \mu_1^u S_{21} + \mu_2^u S_{22} \tag{5.26}$$

$$K_3^u = \mu_1^u + \mu_2^u \tag{5.27}$$

$$\frac{dy}{dx} = \frac{\left(\mu_1^x + \mu_2^x\right)\left(K_1^u\mu_1^{x'} + K_2^u\mu_2^{x'}\right) - \left(K_1^u\mu_1^x + K_2^u\mu_2^x\right)\left(\mu_1^{x'} + \mu_2^{x'}\right)}{K_3^u\left(\mu_1^x + \mu_2^x\right)^2} \tag{5.28}$$

Through the substitution of equation (5.23) into equation (5.21), and the substitution of equation (5.24) into equation (5.28), the complete equations for the first derivatives of the fuzzy model with respect to the input variables u and x result.

The derivatives of the fuzzy model with respect to each of its input variables, u and x, have been derived. It is now possible to apply a gradient descent algorithm to the MPFC structure to optimise the values of the controller output variables, $\mathbf{u}_{1..3}$, for a prediction horizon of 3 steps.

The analysis given is applicable to any number of active fuzzy sets, with the constraint that the fuzzy membership set function and the inference function allow differentiation.

5.4.4. Application of a Gradient Descent Algorithm

This section details the application of a gradient descent method to optimise the controller outputs of the MPFC with respect to a cost function. Initially the example for the MPFC with a prediction horizon of three steps is used. Following this a general equation for a prediction horizon of length **ph** is developed. The reader is referred to Figure 5.36, in Section 5.4.1 of this chapter, where the block diagram of the MPFC for a prediction horizon of 3 steps is illustrated.

The cost function used for the optimisation of the input values u_1, u_2 and u_3 is the weighted sum of the squared errors and is given by equation (5.1). A weighted sum is used in order to weight the later predictions, as this is where the controlled variable approaches the setpoint. When using the gradient descent method, the variables to be adapted are updated using equation (5.29), [58].

$$U(k+1) = U(k) - \alpha \frac{\partial E_p(k)}{\partial U(k)} \tag{5.29}$$

where $U = \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix}^T$ is the vector of manipulated variables to be optimised,
α is the learning gain,
E_p is the cost function value and
k is the adaptation iteration variable.

In order to utilise equation (5.29), the partial derivatives of the cost function, E_p, with respect to each of the fuzzy model inputs u_n , (see Figure 5.36), are required. For a prediction horizon, **ph**, of 3 steps these partial derivatives are :

$$\frac{\partial E_p}{\partial u_1}, \frac{\partial E_p}{\partial u_2} \text{ and } \frac{\partial E_p}{\partial u_3} \tag{5.30}$$

The following details the derivation of the three required partial derivatives.

$$\frac{\partial E_p}{\partial u_3} = \frac{\partial E_p}{\partial y_3}\frac{\partial y_3}{\partial u_3}$$

$$\frac{\partial E_p}{\partial u_3} = \beta_3[y_3 - d_3]FM'(u_3, y_2)_{y_2}$$

$$\frac{\partial E_p}{\partial u_3} = -FM'(u_3, y_2)_{y_2}\beta_3 e_3 \tag{5.31}$$

$$\frac{\partial E_p}{\partial u_2} = \frac{\partial E_p}{\partial y_2}\frac{\partial y_2}{\partial u_2} \tag{5.32}$$

$$\frac{\partial E_p}{\partial y_2} = \frac{\partial}{\partial y_2}\left[0.5\left(\beta_2(d_2 - y_2)_2^{\,2} + \beta_3(d_3 - y_3)^2\right)\right] \tag{5.33}$$

$$y_3 = FM(u_3, y_2) + y_2 \tag{5.34}$$

Substituting (5.34) into (5.33) gives equation (5.35).

$$\frac{\partial E_p}{\partial y_2} = \frac{\partial}{\partial y_2}\left[0.5\left(\beta_2 e_2^{\,2} + \beta_3(d_3 - [FM(u_3, y_2) + y_2])^2\right)\right] \tag{5.35}$$

$$\therefore \frac{\partial E_p}{\partial y_2} = \beta_2[y_2 - d_2] + \beta_3[y_3 - d_3]\left[1 + FM'(u_3, y_2)_{u_3}\right] \tag{5.36}$$

Substituting (5.36) into (5.32) gives :

$$\therefore \frac{\partial E_p}{\partial u_2} = \left[\beta_2(y_2 - d_2) + \beta_3[y_3 - d_3]\left[1 + FM'(u_3, y_2)_{u_3}\right]\right]FM'(u_2, y_1)_{u_2} \tag{5.37}$$

Finally,

$$\therefore \frac{\partial E_p}{\partial u_2} = -FM'(u_2, y_1)_{y_2}\left[\beta_2 e_2 + \beta_3 e_3\left(1 + FM'(u_3, y_2)_{u_3}\right)\right] \tag{5.38}$$

$$\frac{\partial E_p}{\partial u_1} = \frac{\partial E_p}{\partial y_1}\frac{\partial y_1}{\partial u_1} \tag{5.39}$$

$$y_2 = FM(u_2, y_1) + y_1 \tag{5.40}$$

$$\frac{\partial E_p}{\partial y_1} = \frac{\partial}{\partial y_1}\left[0.5\left(\beta_1(d_1 - y_1)^2 + \beta_2(d_2 - y_2)^2 + \beta_3(d_3 - y_3)^2\right)\right] \tag{5.41}$$

Substitution of equations (5.40) and (5.34) into equation (5.41), gives :

$$\frac{\partial E_p}{\partial y_1} = \frac{\partial}{\partial y_1}\left[0.5\left(\begin{array}{l}\beta_1 e_1^2 + \beta_2(d_2 - [y_1 + FM(u_2, y_1)])^2 + \\ \quad \beta_3(d_3 - [y_1 + FM(u_2, y_1) + FM(u_3, y_1 + FM(u_2, y_1))])^2\end{array}\right)\right] \tag{5.42}$$

or

$$\begin{aligned}\frac{\partial E_p}{\partial y_1} &= \beta_1(y_1 - d_1) + \beta_2(y_2 - d_2)\left(1 + FM'(u_2, y_1)_{u_2}\right) \\ &\quad + \beta_3(d_3 - y_3)\left(1 + FM'(u_2, y_1)_{u_2} + FM'(u_3, y_2)_{u_3}\left(1 + FM'(u_2, y_1)_{u_2}\right)\right)\end{aligned} \tag{5.43}$$

Finally,

$$\therefore \frac{\partial E_p}{\partial u_1} = -FM'(u_1, y_0)_{u_1}\left\{\begin{array}{l}\beta_1 e_1 + \beta_2 e_2\left(1 + FM'(u_2, y_1)_{u_2}\right) + \\ \quad \beta_3 e_3\left(\left[1 + FM'(u_2, y_1)_{u_2}\right]\left[1 + FM'(u_3, y_2)_{u_3}\right]\right)\end{array}\right\} \tag{5.44}$$

All of the partial derivatives required for the application of the gradient descent algorithm have been derived and are given by equations (5.31), (5.38) and (5.44).

By means of induction a general equation for the derivative of the error of the MPFC, E_P, with respect to the manipulated variables u_n over the prediction horizon of length *ph* can be deduced. This equation is given by (5.45).

$$\frac{\partial E_P}{\partial u_i} = -FM_i'\left[\beta_i e_i + \sum_{j=i+1}^{ph}\beta_j e_j \prod_{l=j}^{i+1}(1 + FM_l')\right] \tag{5.45}$$

where FM_i' is the partial derivative of the fuzzy model for the prediction step *i* with respect to the manipulated variable, u_i , and
ph is the number of steps in the prediction horizon

5.4.5. Summary of the MPFC Paradigm

Equation (5.45) extends the SPFC to a multi-step predictive controller. Through use of the supervised adaptive fuzzy model strategy developed in Section 4.9, the MPFC could be made adaptive, with the fuzzy model of the plant adapting on-line.

It is beyond the scope of this research to investigate the algorithm suggested for the multi-step predictive fuzzy controller. Two main points would, however, be important in keeping the processing time of the algorithm at a realistic level :

- Initial values for the manipulated variables u_i can be chosen based on user knowledge of the system.

- the optimisation algorithm should have some mechanism to overcome the local minima in the error surface that are a result of the non-linearity of a fuzzy model. A gradient descent method with a momentum term and perhaps some adaptation of the learning gain, α, could help to ensure fast convergence of the algorithm to a set of predicted controller outputs over the prediction horizon.

For application to the warm water process, the fuzzy models of the mass flow and thermal behaviour used in the SPFC in this chapter and in Chapters 6, could be used, whereby the derivatives of the two dimensional fuzzy model derived in this chapter would need to be extended to that of a four dimensional rulebase, thus corresponding to the size of the thermal behaviour fuzzy model of the warm water process.

5.5. Self-Tuning PI Control

5.5.1. Introduction

This section describes the design and simulation of self-tuning PI controllers for the SISO control of the outlet flow, outlet temperature and the multivariable control of the warm water process.

The self-tuning PI controller is based on an algorithm for a self-tuning PID controller of a SISO system by Banyasz and Keviczky [111]. This research contributes the following to the original algorithm :

- the modification of the self-tuning PID controller to that of a self-tuning PI controller and

- the addition of an anti-integral windup mechanism and limits for the PI-controller parameters.

5.5.2. The Mathematical Background of the Self-Tuning PI Controller

The self-tuning PI controllers used to control both the outlet flow and temperature variables of the warm water process are based on an algorithm by Banyasz and Keviczky [111]. This *original algorithm* uses a recursive least squares (RLS) technique to estimate a second order ARX model of the plant to be controlled. The delay between the model input and output must be known a priori. The estimated second order ARX model coefficients are then used to calculate the parameters of a PID controller where the controller zeros cancel the process poles. In contrast to the original algorithm, this research uses a *first* order ARX model of the plant, the coefficients of which, are estimated by a RLS algorithm. The PI controller parameters are calculated so that the PI controller zero cancels the first order process pole. The mathematical derivation of this self-tuning PI controller algorithm, now follows.

The incremental form of an algorithm for a discrete time PI controller is shown by (5.46) - where n serves as the iteration variable.

$$u(n)-u(n-1)=q_0e(n)+q_1e(n-1) \tag{5.46}$$

If a backward difference discretisation method is used, then the coefficients q_0, and q_1 have the following values (5.47,48).

$$q_0 = K_P(1+\frac{T_S}{T_I}) \tag{5.47}$$

$$q_1 = -K_P \tag{5.48}$$

where T_S is the *sampling period*,
K_P is the *Proportional Gain* of the PI controller and
T_I is the *Integral Time Constant* of the PID controller.

This algorithm gives an accurate representation of an ideal PI controller for small sampling times and thus, if desired, the values of the q_0 and q_1 can be calculated directly from analogue PI controller settings [99].

A process model of the form given by equation (5.49) is assumed, i.e. a first order ARX model (in the z domain).

$$\frac{Y(z)}{X(z)}z^{-nk}=\frac{bz^{-(nk+1)}}{1+az^{-1}} \tag{5.49}$$

Letting the controller zero cancel the process pole gives (5.50).

$$1+az^{-1}=1+\frac{q_1}{q_0}z^{-1} \tag{5.50}$$

The forward path transfer function then becomes (5.51) :

$$Q(z)=\frac{q_0bz^{-1}}{1-z^{-1}}z^{-nk} \tag{5.51}$$

This is a system consisting of controller and plant with the open loop transfer function given by equation (5.51). The following analyses the magnitude of the frequency response $|Q(\omega)|$ and the phase response $\angle Q(\omega)$ of this system.

$$|Q(\omega)|=\frac{q_0b}{\left|1-e^{-j\omega T}\right|}$$

$$=\frac{q_0b}{|1-\cos(\omega T)+j\sin(\omega T)|}$$

$$=\frac{q_0b}{\sqrt{4\sin^2(\frac{\omega T}{2})}}$$

At the gain crossover frequency, $|Q(\omega)|=1$, resulting in equation (5.52).

$$q_0b=2\left|\sin(\frac{\omega_0T}{2})\right|\approx 2\frac{\omega_0T}{2}=\omega_0T \tag{5.52}$$

With a phase margin of ϕ_a, the phase response is calculated as follows.

$$\angle Q(\omega_0)=-\omega_0T(nk+1)-\tan^{-1}(\frac{\sin(\omega_0T)}{1-\cos(\omega_0T)})$$

$$=-\omega_0T(nk+1)-\tan^{-1}(\cot(\frac{\omega_0T}{2}))$$

$$= -\omega_0 T(nk+1) - \left(\frac{\pi}{2} - \frac{\omega_0 T}{2}\right) \tag{5.53}$$

$$\phi_a = \angle Q(\omega_0) + \pi$$

$$\therefore 2\phi_a = -\omega_0 T(2nk+1) + \pi \tag{5.54}$$

Combining equations (5.52) and (5.54) results in equation (5.55).

$$q_0 b = \frac{\pi - 2\phi_a}{2nk+1} \tag{5.55}$$

If a phase margin of $\phi_a = \frac{\pi - 1}{2} \approx 60^o$ is applied, then equation (5.56) results.

$$q_0 = \frac{1}{b(2nk+1)} \tag{5.56}$$

From equation (5.50), equation (5.57) is obtained.

$$q_1 = q_0 a \tag{5.57}$$

Equations (5.56) and (5.57) combined with equation (5.47), allow the calculation of the coefficients of the PI controller, K_P and T_I, from the on-line estimated first order ARX model.

A problem often encountered with the PI controller is *integral windup*. One simple solution for integral windup is to stop updating the integral term of the PI controller if the value of *actuator signal* exceeds predefined boundaries, these limits can be chosen to correspond to actuator saturation. To enable the realisation of this solution, the PI algorithm can be rewritten in the form of equation (5.58) where the proportional $u_P(n)$ and the integral $u_I(n)$ terms are calculated separately.

$$u_P(n) = K_P e(n) \qquad \text{Proportional Term}$$

$$u_I(n) = u_I(n-1) + K_P \frac{T_s}{T_I} e(n-1) \qquad \text{Integral Term}$$

$$u(n) = u_P(n) + u_I(n) \tag{5.58}$$

The PI controller algorithm described by equations (5.46), (5.47), (5.50), (5.56), (5.57) and (5.58) is used as a basis for a self-tuning PI controller.

5.5.3. Simulation Software for the Self-Tuning PI Controller

This section describes the simulation software developed for the self-tuning PI controller. The simulation icons for the MATLAB/SIMULINK environment are shown in Figure 5.38, see page 161, and are explained below :

- *"RLS ID"* - is the ARX Model RLS identification algorithm. This algorithm has the capability to accept initial values for the estimated coefficients in its user menu. Thus previous system identification results can be used to initialise the ARX model RLS coefficients.

- *"PI Parameters", "PID Parameters"* - calculate the controller parameters for PI and PID controllers respectively, based on the Banyasz and Keviczky method. The user can specify the desired phase margin of the controller response and the maximum and minimum values for the PI/PID controller coefficients. Both of these icons require the ARX model coefficients from the RLS algorithm as inputs.

- *"Adaptive PI", "Adaptive PID"* - are the simulation icons for PI and PID controllers which require the corresponding PI(D) controller parameters and the error signal as inputs.

- *"Adapt. with Anti-Windup"* - is the simulation icon for a PI controller with ant-integral windup, where the integral term update depends on a user defined value of the magnitude of the error signal.

- "*Adapt. PI with u-fback"* - is the simulation icon for a PI controller with anti-integral windup where the integral update is only performed when the actuator signal lies within user defined boundaries. The fourth input is for the actuator signal.

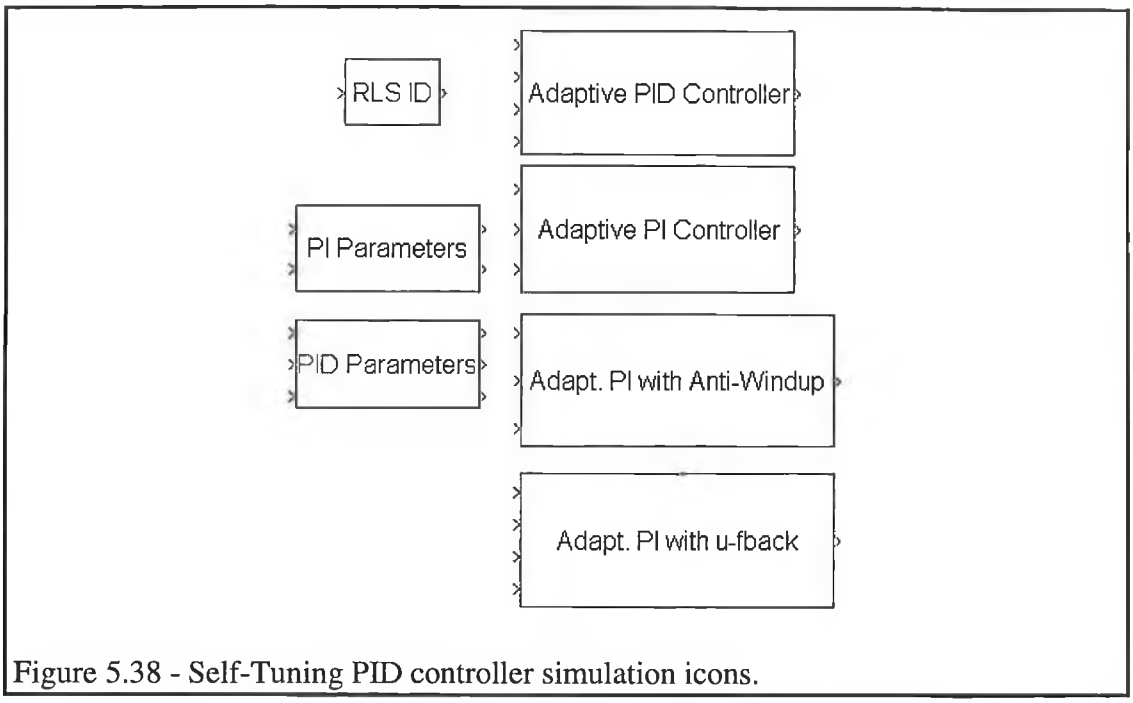


Figure 5.38 - Self-Tuning PID controller simulation icons.

As the warm water process variables to be controlled, have non-zero steady state values, high pass filters where applied to all of the inputs of the RLS identification algorithm. These high pass filters serve to remove DC signals from the inputs and are discrete IIR fourth order Butterworth filters with a cut-off frequency of 0.000333 Hz. This frequency is equivalent to one hundredth of the sampling frequency, where the sampling period is 30 seconds.

5.5.4. Results

5.5.4.1. Introduction

This section of the chapter presents the results of the simulated self-tuning PI control of the MISO and multi-variable control of the outlet flow and temperature variables of the warm water process.

The ANN model of the warm water process was used as the plant model in all of the self-tuning PI control simulations. The phase margin ϕ_a was set to a value of $\frac{\pi - 1}{2}$ radians, corresponding to approximately 60 degrees. For the chosen phase margin, the controller zero should cancel the process pole resulting in a first order forward loop transfer function, thus no overshoot should occur in the controller response.

For all simulations in this section, the value of the forgetting factor used in the RLS identification algorithm was set to 0.99. This value was found to give a good compromise between the robustness and adaptability of the RLS algorithm.

For the MISO and multivariable control of the warm water process, static decoupling of the plant was utilised. Static decoupling attempts to decouple the controlled variables at DC frequency. The values of the elements of this decoupling matrix were calculated from the first order ARX models of the mass flow and thermal behaviour of plant as described Section 4.8 of this thesis. The decoupling matrix is calculated from the general modelling equation of the plant, equation (5.59), as follows :

$$\begin{bmatrix} F_{out}(q) \\ T_{out}(q) \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} \begin{bmatrix} F_{cold}(q) \\ F_{hot}(q) \end{bmatrix} \tag{5.59}$$

where M_{11}, M_{12}, M_{21} and M_{22} are ARX models.

Calculating the steady state gains of the individual models and inverting (5.59) gives :

$$\begin{bmatrix} F_{hot}(q) \\ F_{cold}(q) \end{bmatrix} = M \begin{bmatrix} F_{out}(q) \\ T_{out}(q) \end{bmatrix} \tag{5.60}$$

where $\mathrm{M} = \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix}^{-1}$ is the decoupling matrix at steady state plant and

K_{11}, K_{12}, K_{21} and K_{22} are the steady state gains of the ARX models M_{11}, M_{12}, M_{21} and M_{22}.

The operating point chosen for this matrix corresponded to an outlet flow of 200ml/s and thus the ARX model coefficients were taken directly from Table 4.5 on page 89. This decoupling matrix is shown below :

$$\text{Steady state decoupling matrix, } \mathrm{M} = \begin{bmatrix} 0.73685 & 2.83872 \times 10^{-5} \\ 0.60327 & -2.83872 \times 10^{-5} \end{bmatrix}$$

5.5.4.2. ST-PI Outlet Flow Control

Figure 5.39 shows the block diagram of the software used to simulate the ST-PI control of the outlet flow of the warm water process. The first outlet flow ST-PI control simulation was performed *without initialisation of the first order ARX model coefficients* in the RLS identification block. The setpoint for the outlet flow was 100 ml/s with a square wave offset of ±12.5 ml/s. The results of this first simulation are shown in Figures 5.40 - the outlet and inlet flows, 5.41 - the identified ARX model coefficients and 5.42 - the PI controller constants.

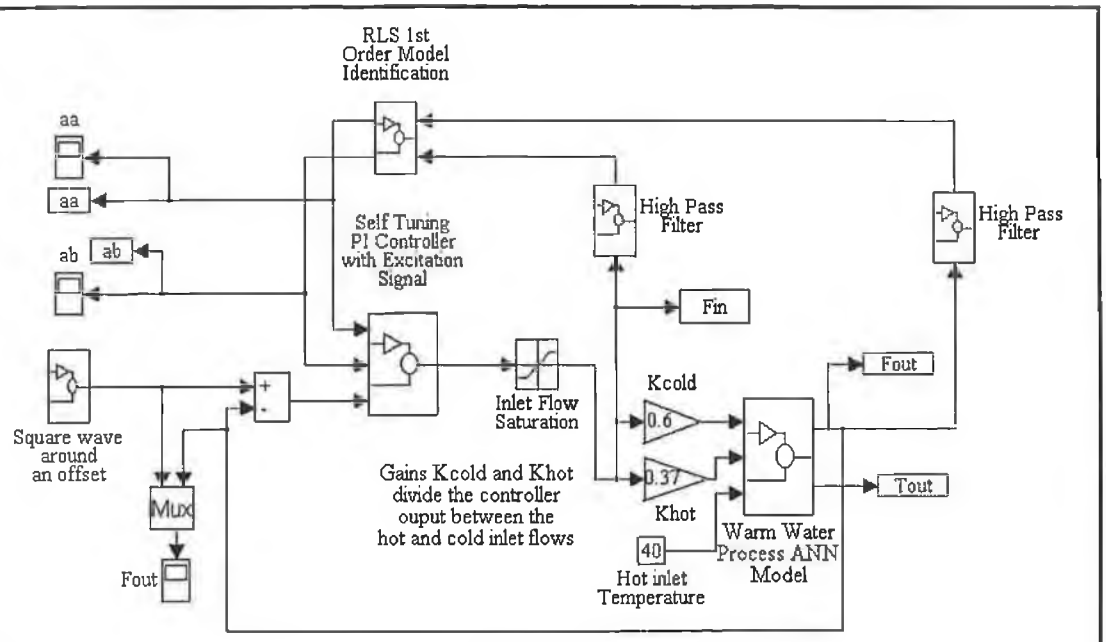


Figure 5.39 - Block diagram of the self-tuning PI control of the outlet flow.

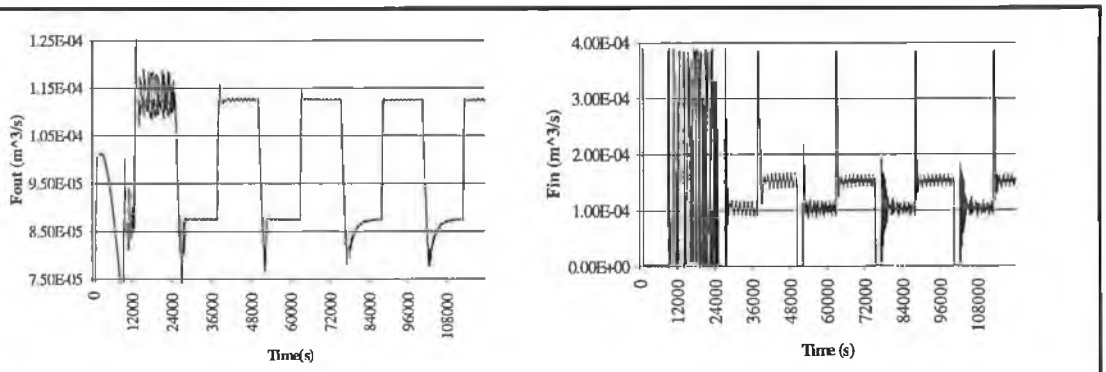


Figure 5.40 - The inlet and outlet flows for *uninitialised* ST-PI outlet flow control.

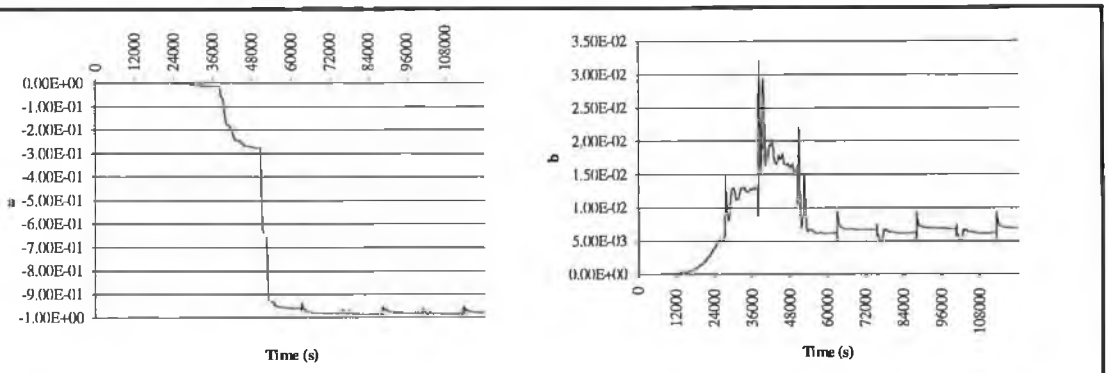


Figure 5.41 - ARX model coefficients for *uninitialised* ST-PI outlet flow control.

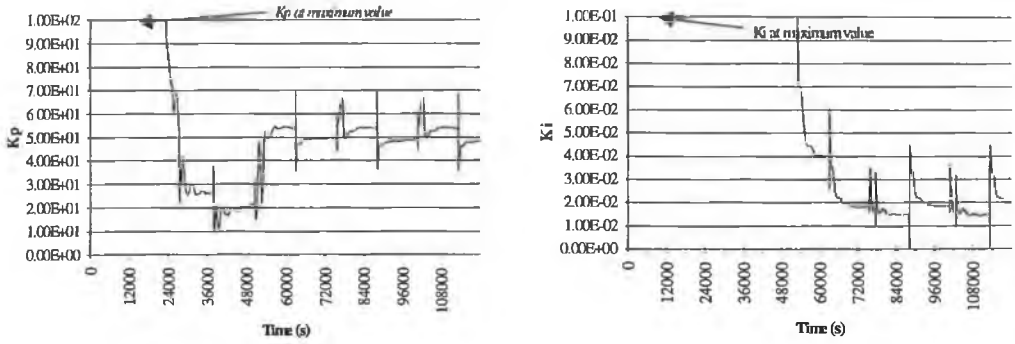


Figure 5.42 - The PI controller parameters for *uninitialised* ST-PI outlet flow control.

The self-tuning PI controller is able to control the outlet flow of the warm water without any steady state error and with a rise time of approximately 1000 seconds around the operating point of 100 ml/s. The overshoot present on the negative going steps is due to the fact that the PI controller zero does not cancel the first order process pole and thus a second order forward loop transfer function results. This implies that the first order ARX model of the process is not correct at this operating point. The incorrectly identified ARX model is due to the non-linearity of the process around the chosen operating point.

Without initialisation, the ARX model coefficients converge only after the equivalent of 7200 samples. By means of limiting the PI controller parameters to user defined maximum values, the controller performance is not adversely effected before ARX model conversion but strong variance is evident on the manipulated variable. If the PI controller parameter limits were not present, then small or zero values of the ARX model coefficients would result in unrealistically large PI controller parameter values, thus leading to adverse controller performance. The spikes evident in the estimated ARX model coefficients are caused by the step signal on the setpoint.

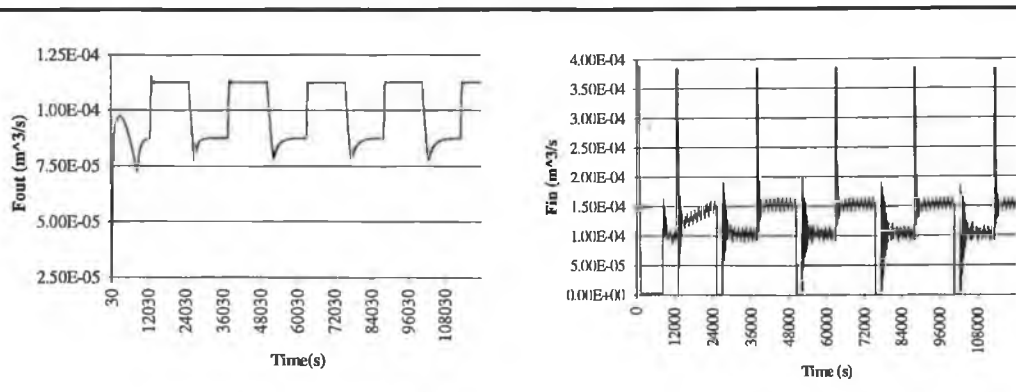


Figure 5.43 - The inlet and outlet flows for initialised ST-PI outlet flow control.

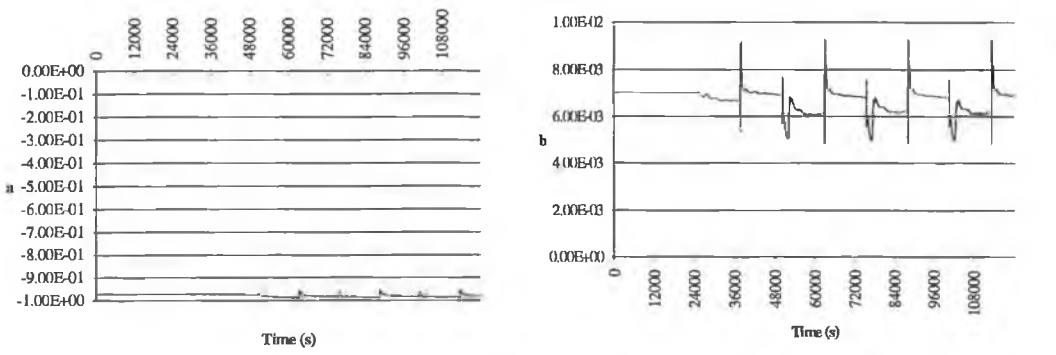


Figure 5.44 - ARX model coefficients for *initialised* ST-PI outlet flow control.

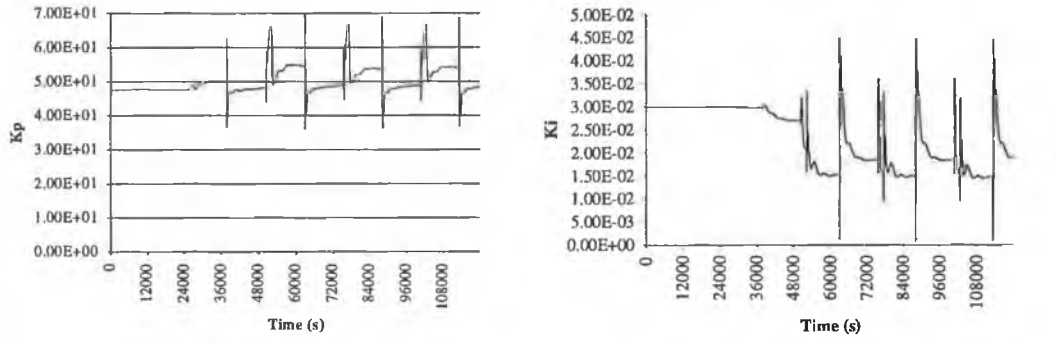


Figure 5.45 - PI controller constants for *initialised* ST-PI outlet flow control.

The variance of the input flow signal evident after convergence is attributable to the excitation signal of amplitude ±10 ml/s, which is added to the PI controller output to assist the RLS identification algorithm to learn the system transfer function in closed loop.

The second simulation performed is a repeat of the previous simulation with initialised first order ARX model coefficients. The initial values used are taken from converged values of the previous simulation and are - a = -0.97 and b = 0.0085. Due to the ARX model coefficient initialisation, the initial controller performance is better than the previous simulation. This improved performance is clearly seen in the reduced inlet flow variance during the first 20000 seconds of operation. The controller dynamics are very similar to those of the uninitialised controller. The results of this simulation are found in Figures 5.43 - the outlet and inlet flows, 5.44 - the identified ARX model coefficients and 5.45 - the PI controller constants.

Figure 5.46 shows the responses of the ST-PI controller around three other operating points - 200 ml/s ±25 ml/s, 150 ml/s ±18.75 ml/s and 50 ml/s ±6.25 ml/s. These results show that the ST-PI can adapt to different operating points and still provide good control. The differing dynamic responses of the controller reflect the plant

dynamics at the different operating points. Due to plant non-linearities, the ARX model identification is not correct as second order responses are clearly seen at 150 ml/s and 200 ml/s.

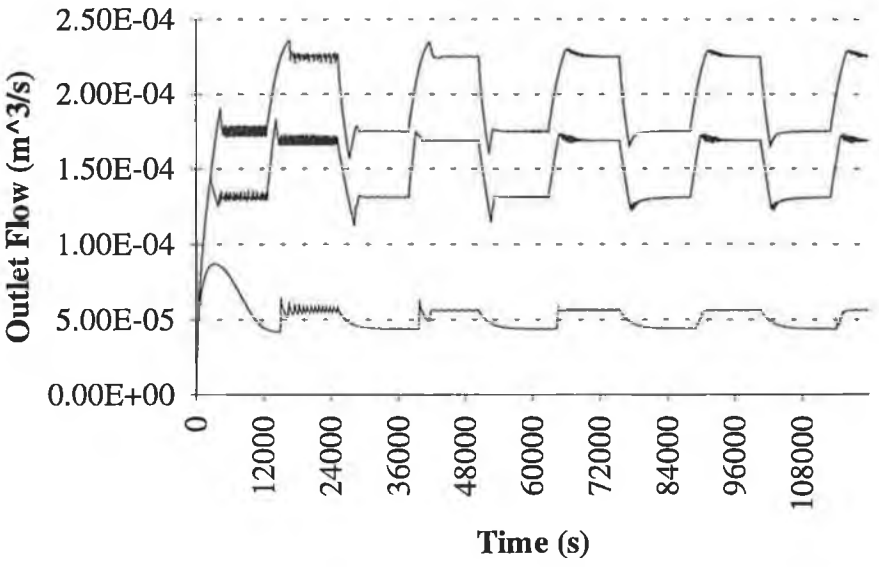


Figure 5.46 - ST-PI outlet flow control around three operating points.

5.5.4.3. ST-PI Outlet Temperature Control

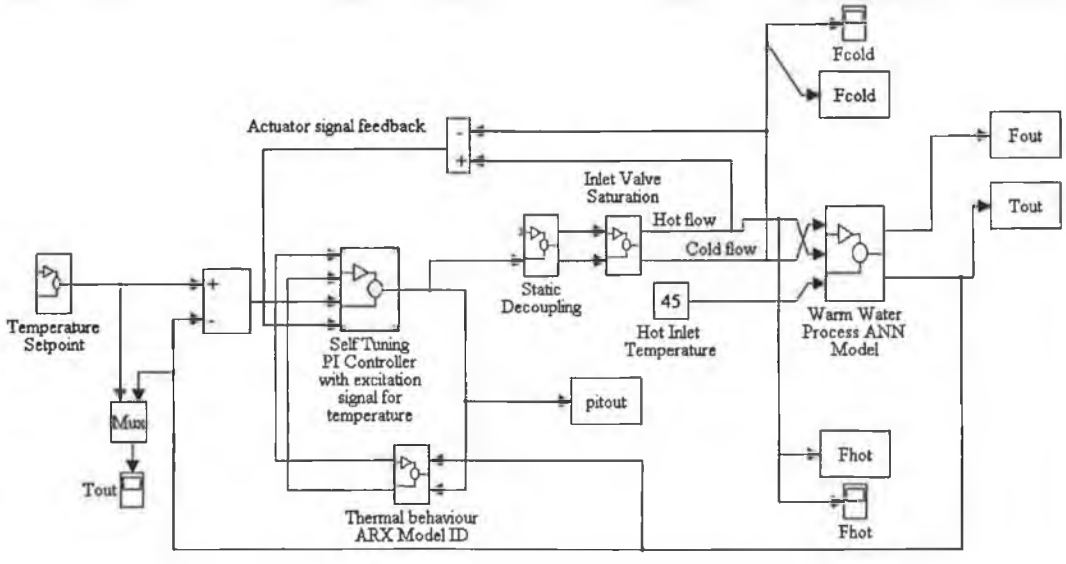


Figure 5.47 - Self-tuning PI control of the warm water process outlet temperature.

This section details two simulations of ST-PI control of the outlet temperature of the warm water process, where Figure 5.47 shows the simulation software used. The first

simulation assumes that no disturbances are present on the hot and cold inlet temperatures. In order to investigate the disturbance rejection properties of the ST-PI, the second simulation adds sinusoidal disturbance signals to the hot and cold inlet temperature variables as defined by equation (5.7) and used in the SPFC temperature control in Section 5.3.

The results of the simulation without disturbance variables are shown in Figures 5.48 - the outlet temperature and hot and cold inlet flows, 5.49 - the identified ARX model coefficients and 5.50 - the PI controller constants. The oscillatory response of the controller is due to the following causes :

- the inherent non-linearity of the plant thermal behaviour and
- osillatory nature of the manipulated variables.

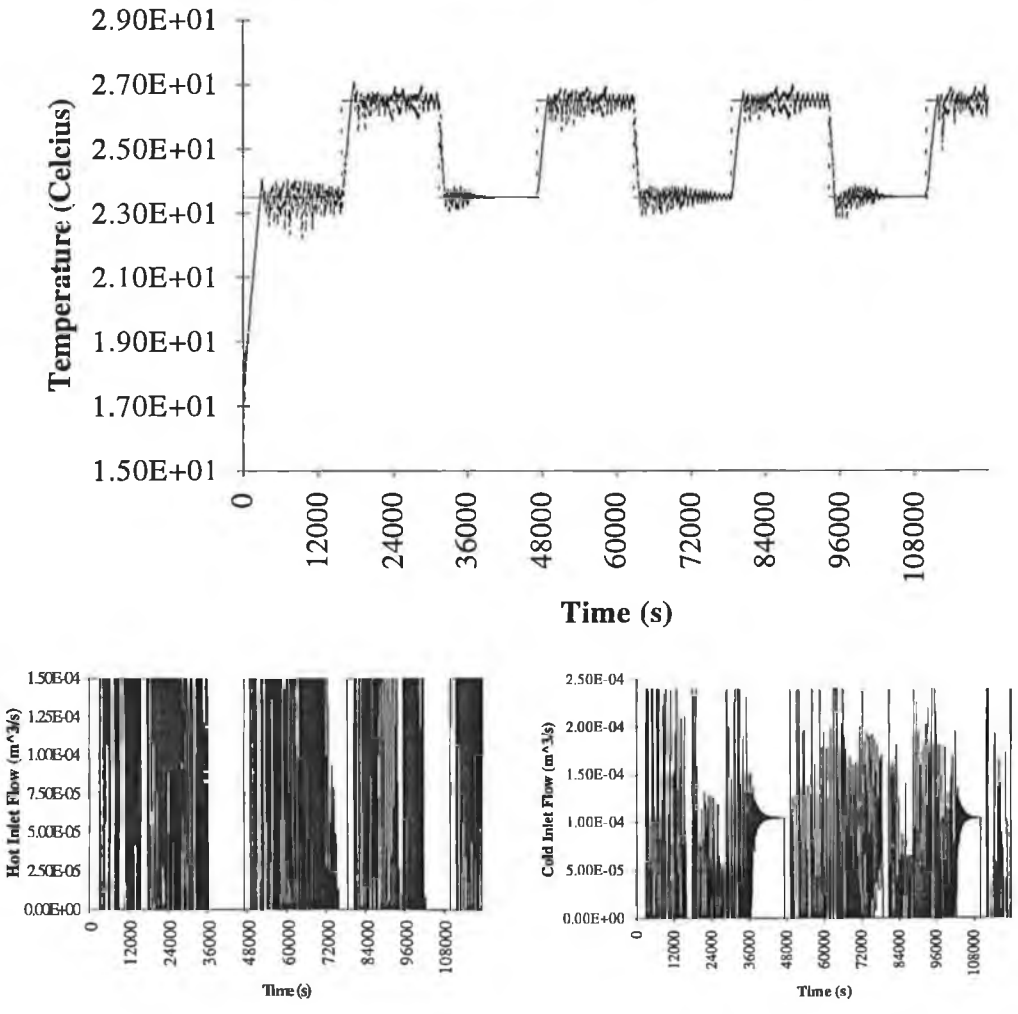


Figure 5.48 - Outlet temperature and inlet flows for ST-PI control of outlet temperature.

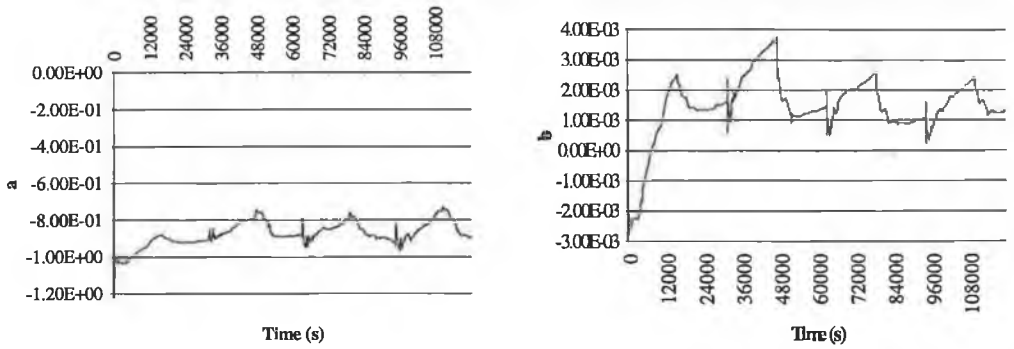


Figure 5.49 - Identified ARX coefficients for ST-PI outlet temperature control.

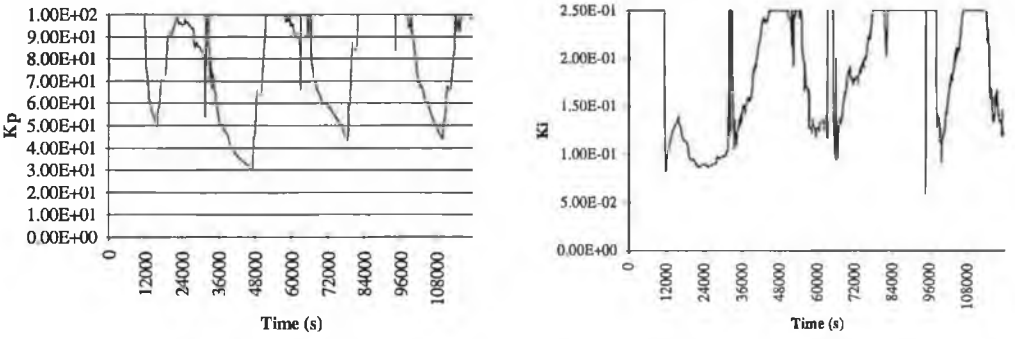


Figure 5.50 - PI Controller coefficients for ST-PI outlet temperature.

Even though the response is oscillatory, the ST-PI controller is able to follow the setpoint with an accuracy of ±0.5 degrees Celsius after settling time, corresponding to 2% of full scale. Considering the non-linearity of the thermal response of the warm water process, this error is quite satisfactory. The variance on the manipulated variables is, however, more similar to an on-off controller and when run on a real system could lead to acutator wear and eventual failure.

Figure 5.51 shows the response of the ST-PI controller with a disturbance signal added to the hot inlet temperature. The disturbance signal used is that applied to the SPFC for outlet temperature control and defined by equation (5.7) on page 136. There is practically no difference between the non-disturbed and disturbed controller responses. This good disturbance rejection is attributable to the direct feed back in the ST-PI controller design.

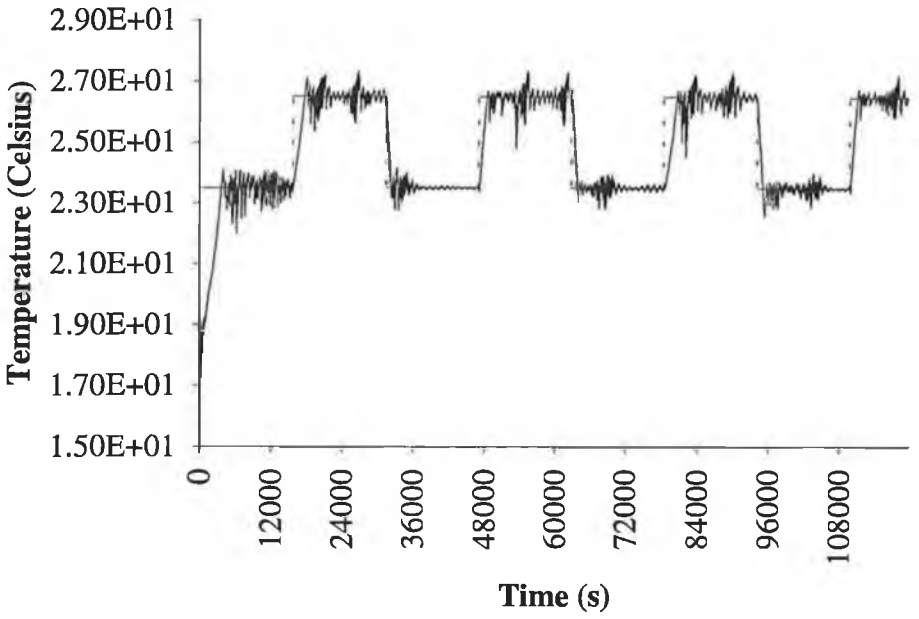


Figure 5.51 - Temperature control with disturbance variables.

Figure 4.52 contains the ST-PI controller response for the outlet temperature setpoints of 20 ±2.5, 25 ±2.5 and 30 ±2.5 degrees Celsius. The ST-PI controller achieves the same level of outlet temperature control for these three operating points with oscillation of the controller variable being due to the non-linearity of the plant.

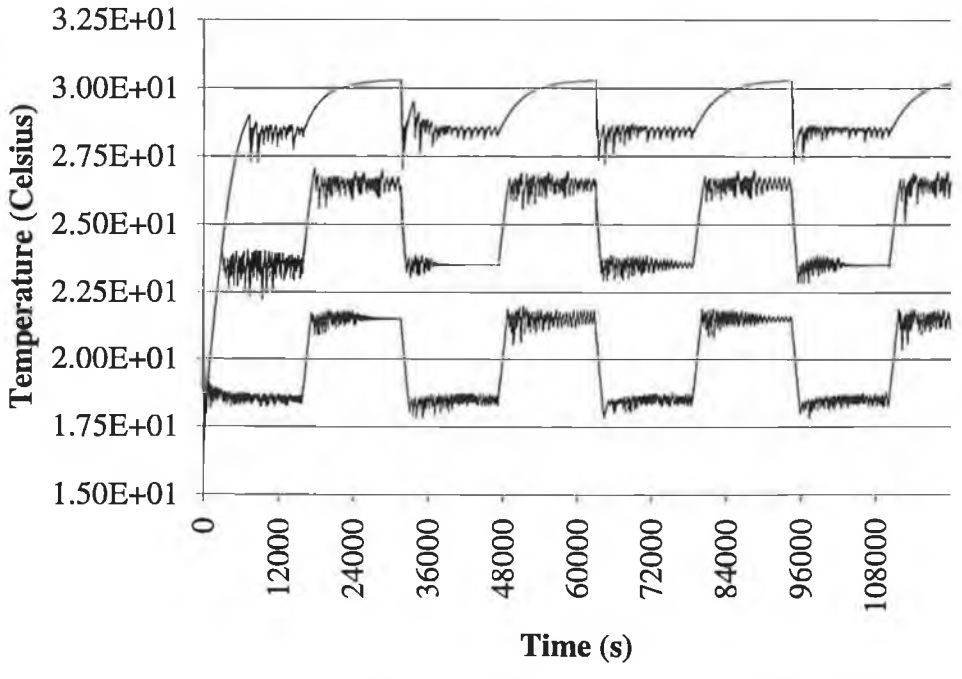


Figure 5.52 - ST-PI outlet temperature control around three operating points.

5.5.4.4. ST-PI Multivariable Control

Two examples of the mulitvariable ST-PI control of the outlet flow and temperature of the warm water process are briefly presented in this section. A block diagram of the simulation software used for these simulations is shown in Figure 5.53.

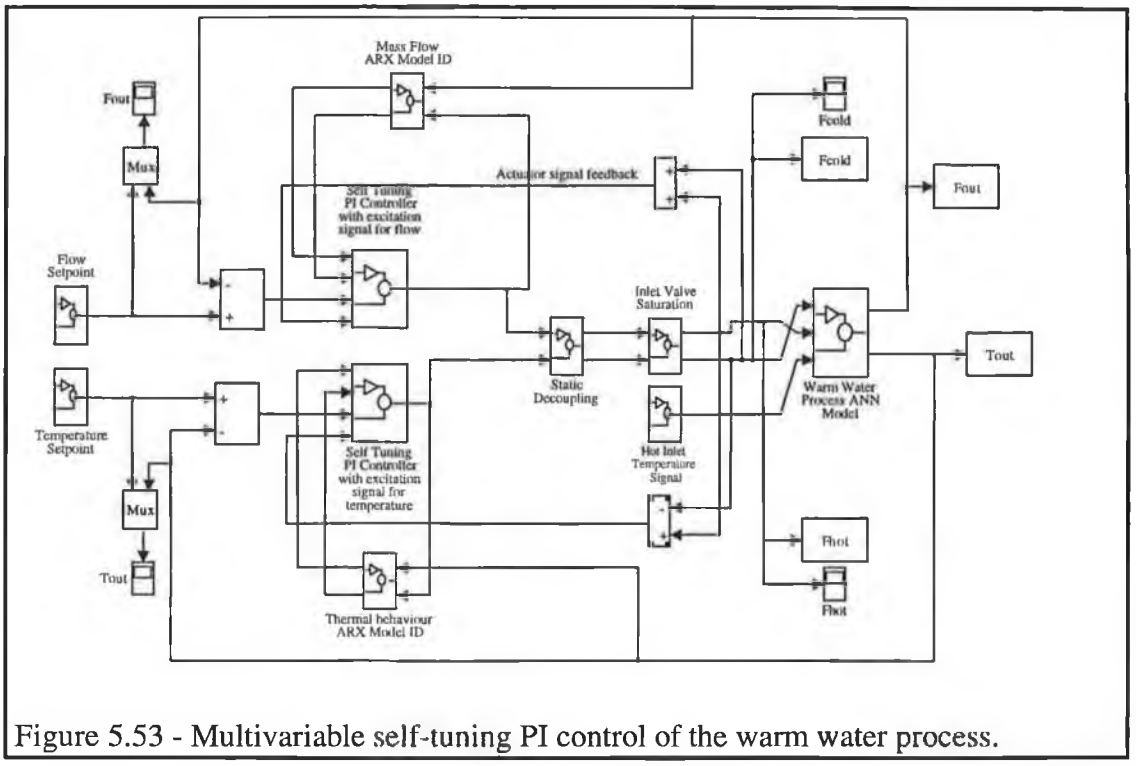


Figure 5.53 - Multivariable self-tuning PI control of the warm water process.

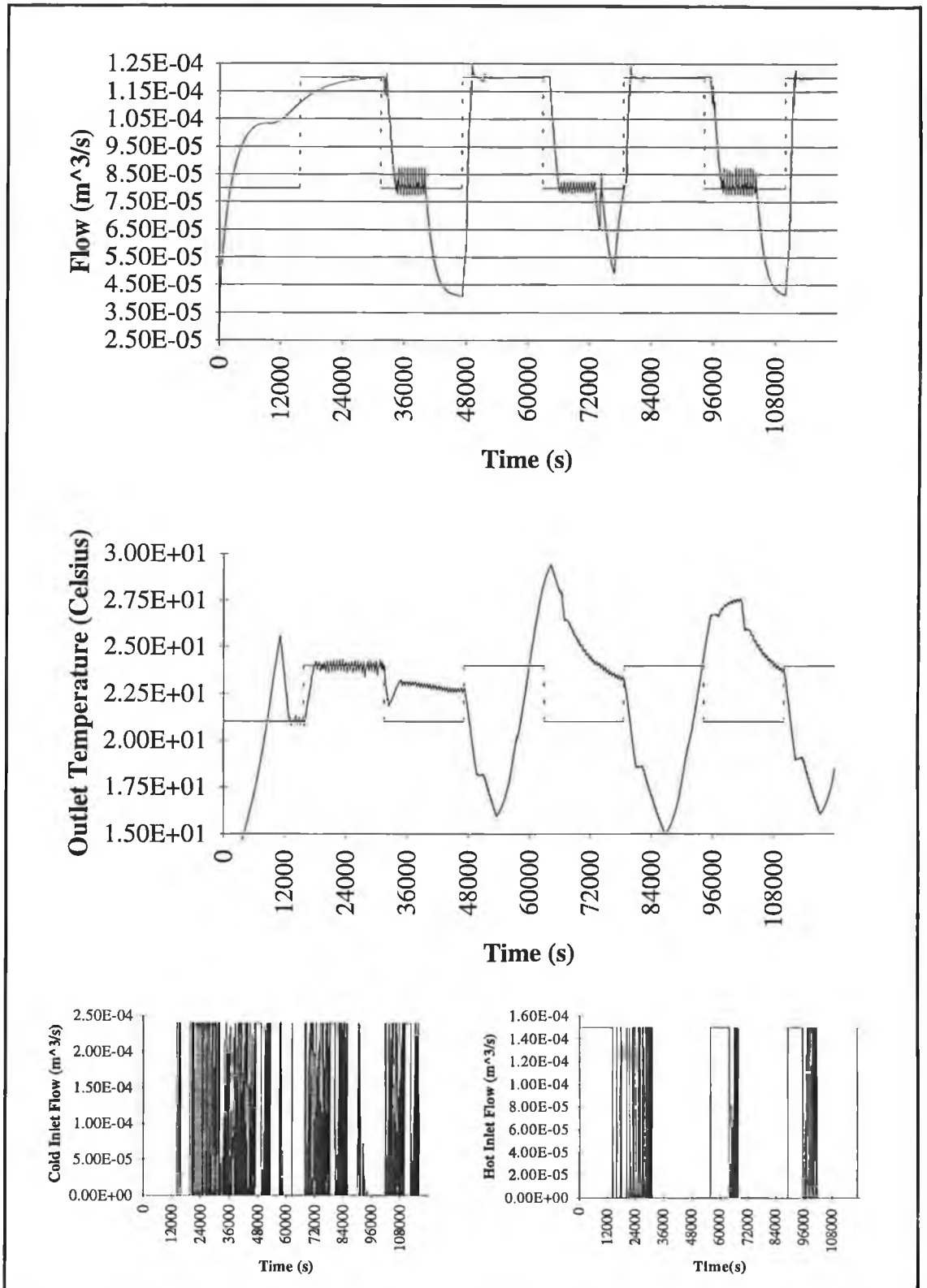


Figure 5.54 - Multivariable ST-PI control - setpoints 22.5 ±1.5 degrees and 100 ±20 ml/s *without* hot inlet temperature disturbance signal.

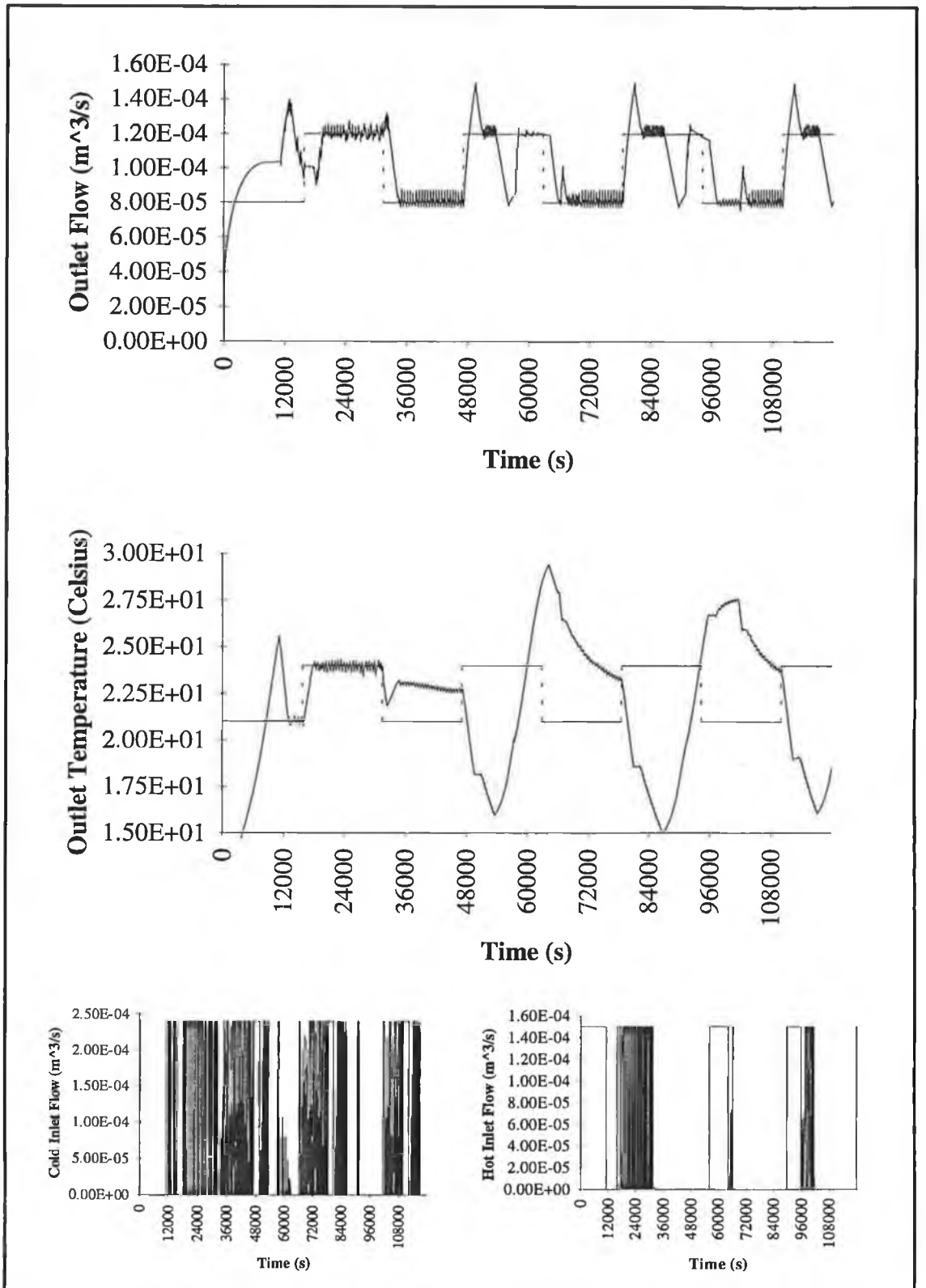


Figure 5.55 - Multivariable ST-PI control - setpoints 22.5 ±1.5 degrees and 100 ±20 ml/s *with* hot inlet temperature disturbance signal.

As the ST-PI controller is a SISO controller, the decoupling matrix described in Section 5.4.4.1 and given by equation (5.60) is used to decouple the controlled

variables. By means of decoupling, separate SISO ST-PI controllers for both outlet flow and temperature can be utilised. The decoupling matrix used, is for DC frequency decoupling at an operating point of 200 ml/s.

The first simulation, without an added disturbance signal on the hot inlet flow temperature, is shown in Figure 5.54 and the controller response with a disturbance hot inlet flow temperature is shown in Figure 5.55. Both controller responses show that the ST-PI controller with the static decoupling is unable to control both outlet flow and outlet temperture concurrently. This is due to the fact that the decoupling used is a simple matrix which does not even attempt to compensate for the non-linear dynamics of the plant. If accurate non-linear dynamic decoupling functions were utilised the ST-PI would exhibit improved multivariable control of the plant.

5.5.5. Summary

This section of the chapter has described the development of a SISO self-tuning PI control strategy and its application to the warm water process for the MISO and multivariable control of the outlet flow and temperature variables. The following section compares this control strategy with the SOC and SPFC paradigms and chooses an adaptive fuzzy control strategy for application to the control of the real plant.

5.6. Controller Appraisal

A comparison and contrast of the three adaptive control strategies described in this chapter follows, with one of the adaptive fuzzy control paradigms being chosen for application to the control of the real warm water process.

- *Controller error* - based on controller error, the best controller from the three controllers is clearly the ST-PI controller, which exhibits zero steady state error for control of the outlet flow. The steady state error performance of the two adaptive fuzzy controllers for outlet flow control was approximately equal, with both exhibiting steady state errors of approximately 2%. In the case of the SPFC, this steady state error of the controller is attributable to the intrinsic modelling error of the fuzzy models used to predict the plant behaviour. This modelling error can be improved by increasing the resolution of the fuzzy models through an increase in rulebase size. For temperature control, the ST-PI delivered superior error performance when compared to the SPFC. The error of the SPFC is again due largely to the intrinsic modelling errors of the fuzzy

model used to predict the thermal behaviour of the plant. Whilst offering the best outlet temperature controller error, the ST-PI controller has the disadvantage of large manipulated variable variance due to the large forward loop gain and, thus, because of actuator wear, would not be suitable for real time plant control. The SPFC does not exhibit large variance in the manipulated variable due to the fact that the user can indirectly define the controller gain through specification of the desired controller response by means of definition of the first order reference model parameters.

- *Disturbance rejection* - due to the direct feedback nature of the ST-PI controller, it offers superior disturbance rejection compared to the SPFC. The poor disturbance rejection of the SPFC is due to the utilisation of the enthalpy term in the fuzzy model. This poor disturbance rejection could be improved by modifying the structure of the fuzzy model so that the hot inlet flow and temperature variables are separate fuzzy model inputs. However, this extension has the disadvantage of substantially increasing the memory requirements of the fuzzy model.

- *Response specification* - Both of the adaptive fuzzy controllers allow some degree of specification of their dynamic response. In the case of the SOC, this specification is by means of a heuristic reference model (often in lookup table format), which does not allow deterministic response specification and resulted in unpredictable controller behaviour with poor overall controller responses. The SPFC uses a first order system as a reference model to specify the response of the controller, which enables clear definition of the controller behaviour and resulted in SPFC responses which generally corresponded to the specified dynamics, when no saturation of the manipulated variables occured. Due to the deterministic nature of the SPFC reference model, it offers the best method of controller response specification of the two adaptive fuzzy controllers.

- *Algorithmic complexity and clarity of design* - of the three adaptive fuzzy controllers, the ST-PI controller is the least complex and most easily understood of the three controller paradigms. Due to the fact that the SPFC is a predictive controller and utilises a clear structure consisting of plant model and reference model, it is more easily understood than the SOC. The SOC is the most complex and confusing of the three adaptive controller designs. This is due largely to the ad-hoc nature of many of its parameters, e.g. the lookup table reference model, and the use of false terminology, e.g. performance index instead of reference model.

- *Convergence and adaptability* - the ST-PI continuously adapts to identify the ARX model coefficients whereby some experimentation with the value of forgetting factor was necessary in order to achieve a good compromise between the robustness and adaptability of the controller. For larger forgetting factor values, the controller requires more time to converge but is more robust to noise whereas smaller values lead to fast convergence with poor noise rejection. The SPFC adapts a global fuzzy model of the plant only when the modelling error is reduced by the suggested rulebase. Due to the global nature of the fuzzy model, large amplitude learning signals are required for the model to learn the plant behaviour around the operating performance. This is clearly a disadvantage when compared to the small amplitude of the learning signal used in the RLS ARX identification. This disadvantage is, however, compensated by the fact that the fuzzy model is a global model of the plant. The adaptation of the SOC rulebase is poorly explained in the literature and can lead to controller instability if the consequent values in the rulebases are not limited. Moreover, the use of the heuristic reference model in lookup table format led to poor overall controller responses.

- *Processing and memory requirements* - the processing and memory requirements of the ST-PI controller are superior to those of both adaptive fuzzy control strategies. The SOC is more efficient than the SPFC in terms of memory and processing requirements. The SPFC is extremely intensive in terms of memory and computational requirements. The time requirements of the SPFC are due to the controlled search algorithm, which can lead to hundreds of evaluations of the fuzzy model. This could be improved through the use of some form of gradient descent algorithm to find the optimal controller output for a given cost function. The prerequisite for such a method is that the fuzzy model be differentiable, where all functions and parameters in the fuzzy model must allow differentiation. Functions such as triangular fuzzy membership functions and the minimum inference function do not fulfill this criteria as they cause discontinuities. The large memory requirements of the fuzzy models used in the SPFC prevent the porting of this algorithm to current microcontrollers. For applications using IBM compatible PCs and workstations these memory requirements are, however, not prohibitive.

Based on the above considerations, the single step fuzzy predictive controller has been selected for evaluation on the real warm water process. The following chapter

presents the results of the control of the outlet flow and temperature variables using the SPFC paradigm.

Chapter 6 - *Real Time Control of the Warm Water Process*

6.1. Introduction

6.1.1. General Introduction

This chapter presents the results and appraisal of the application of the *Single Step Predictive Fuzzy Controller (SPFC)* developed in Chapter 5 to the real warm water process. This controller is used for the *Single Input Single Output (SISO)* control of the outlet flow and the *Multi Input Single Output (MISO)* control of outlet temperature of the warm water process. Due to access restrictions and a series of compressor breakdowns, only a limited amount of time could be spent evaluating the SPFC on the real plant. As a result of these practical restrictions, multivariable control of the real warm water process could not be evaluated.

The *SPFC* used in all experiments in this chapter is the design developed in Chapter 5 which is illustrated in Figure 5.20 and characterised by the following points :

- a *first order reference model* is used to define the dynamic response of the controller, where the user defines the time constant of the reference model.

- the *use of supervised adaptive fuzzy models* to predict plant behaviour.

- as the *ANN plant model* best represents the mass flow and thermal behaviour of the plant, the fuzzy models used to model the mass flow and thermal behaviour of the warm water process in the SPFCs, were initially trained on the ANN model of the warm water process. Figure 5.20 shows the structure of these fuzzy models.

- except where otherwise specified, the *initial outlet flow* for all experiments was zero, corresponding to an empty tank, and the initial outlet temperature was approximately 17 degrees Celsius.

- The *computer* used was an IBM compatible PC with an Intel 386DX 33MHz processor and 4MBytes of RAM.

6.1.2. Overview of Chapter Structure

Sections 6.2. and 6.3. of this chapter present and evaluate the results from a set of experiments for SFPC control of the outlet flow and temperature of the warm water

process respectively. *Section 6.4.* suggest methods to improve the results from the experiments and concludes the chapter.

6.2. Warm Water Process Outlet Flow Control

Using the SPFC, a *set of six experiments* for control of warm water process outlet flow were carried out, all with a sampling time of 30 seconds. The outlet flow rate setpoints of these experiments are : 0.00005 m^3/s, 0.000075 m^3/s, 0.0001 m^3/s, 0.00015 m^3/s, 0.000175 m^3/s and 0.0002 m^3/s. Figures 6.1, 6.2, 6.3, 6.4, 6.5 and 6.6 contain graphs of the inlet and outlet flows for these experiments respectively, where *inlet flow* refers to the sum of the cold and hot inlet flows.

The *SPFC for the outlet flow control* of the warm water process utilised the first order reference model for the desired response with a desired time constant of 130 seconds. As the time constant of the mass flow is approximately 500 seconds at an outlet flow of 100 ml/s, this chosen time constant attempts to deliver control with dynamics which are considerably faster than the real plant.

The following steps are carried out in the outlet flow SPFC for each sample :

- The *supervised adaptive fuzzy model* firstly uses the current data sample from the plant to adapt its rulebase. The adapted rulebase is adopted into the fuzzy model only if the modelling error for the current data sample is reduced, otherwise the old rulebase is kept.

- Following this fuzzy model adaptation, the *first order reference model* calculates the next desired outlet flow based on the current plant outlet flow and setpoint values.

- A *controlled search* is performed of the rulebase in order to determine the controller output which minimises the cost function, i.e. the magnitude of the error between the desired response and that calculated by the fuzzy model for a given value of the manipulated variable. This search is performed by firstly evaluating possible controller outputs at ten equally spaced reference inlet flow values (where inlet flow is the manipulated variable) across the full scale of 0 to 250 ml/s, and then performing a fine search around the best three of these ten reference points.

- The *value of inlet flow* which results in the lowest value of the cost function, is then used as the controller output, whereby the inlet flow value is divided

equally into setpoints for the hot and cold inlet flows. In order to reduce the possibility of an overflow, the inlet flow value of the warm water process is limited to a value of 250 ml/s, which results in a steady state tank level of approximately 180 cm.

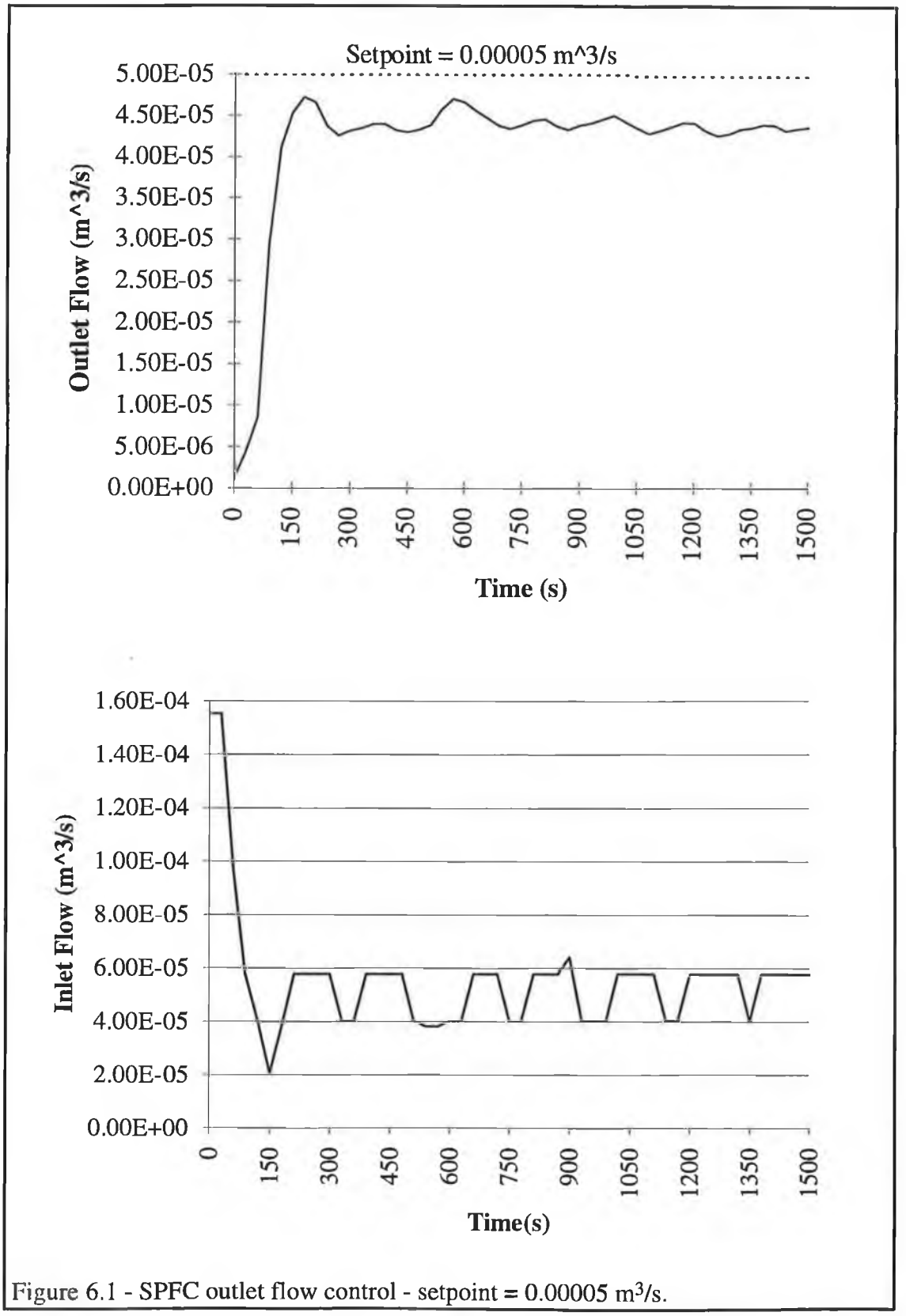
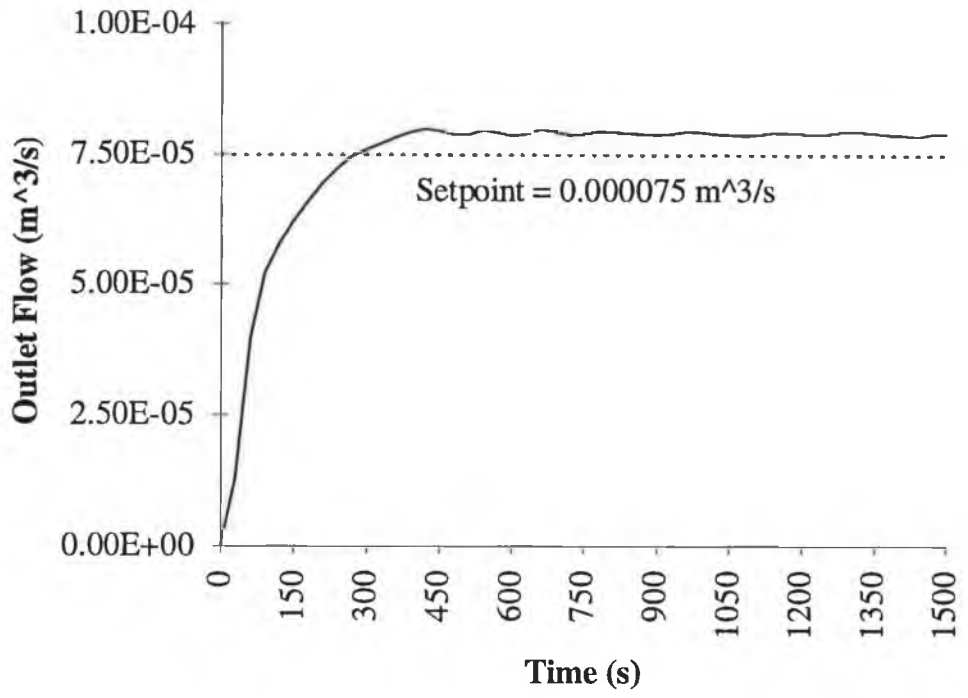


Figure 6.1 - SPFC outlet flow control - setpoint = 0.00005 m^3/s.



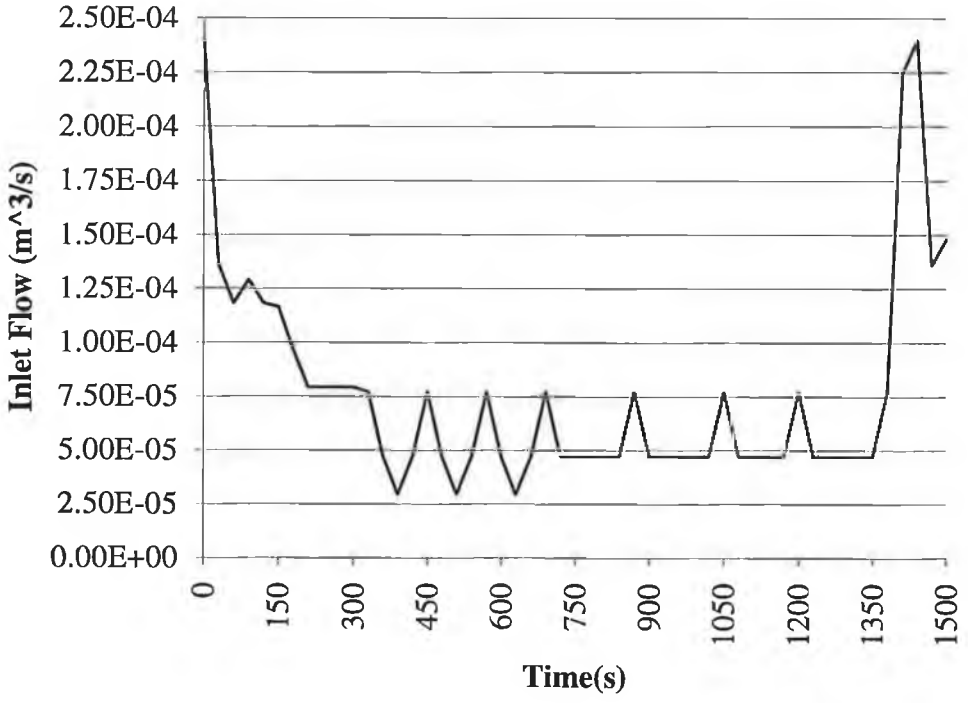


Figure 6.2 - SPFC outlet flow control - setpoint = 0.000075 m^3/s.

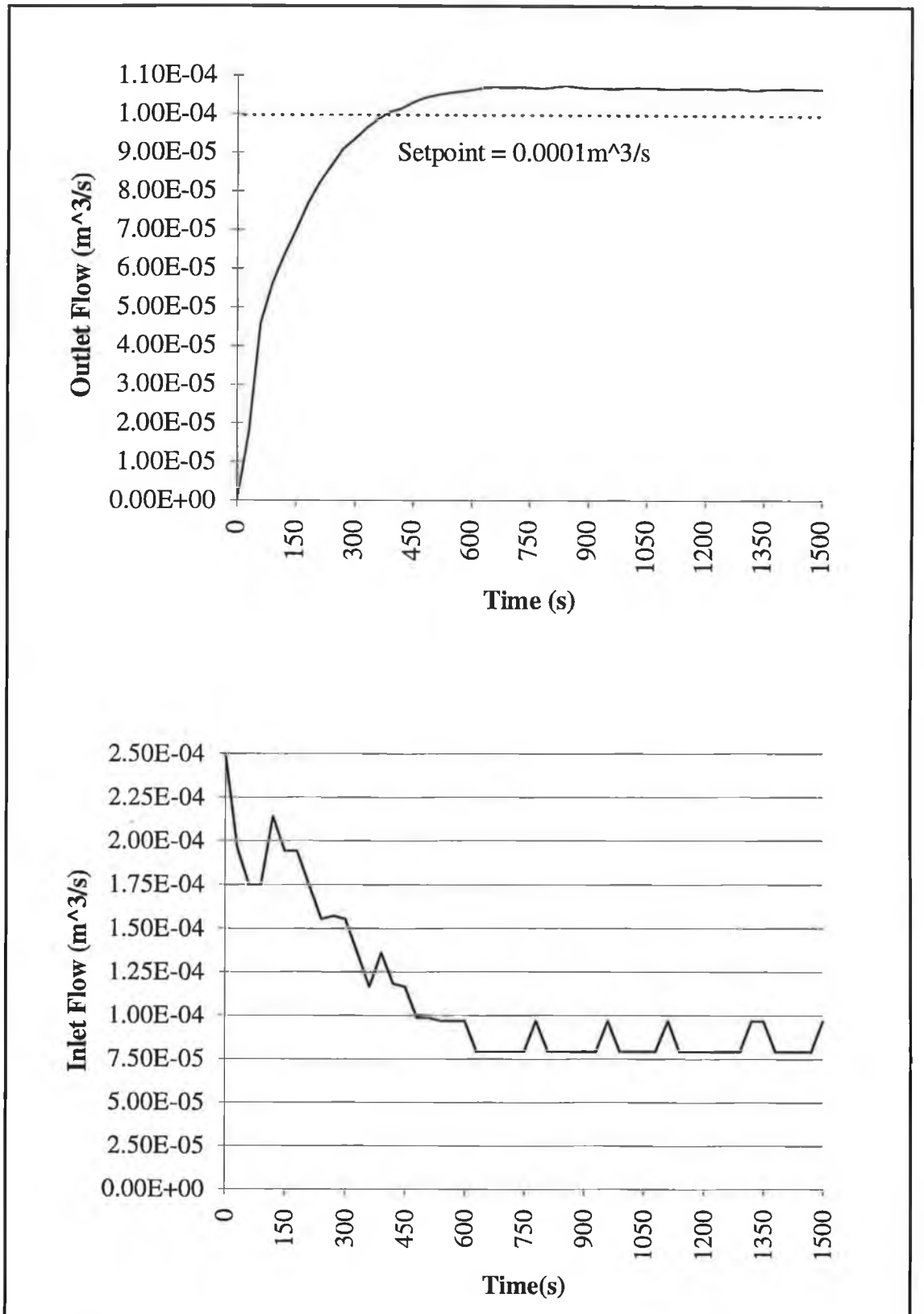
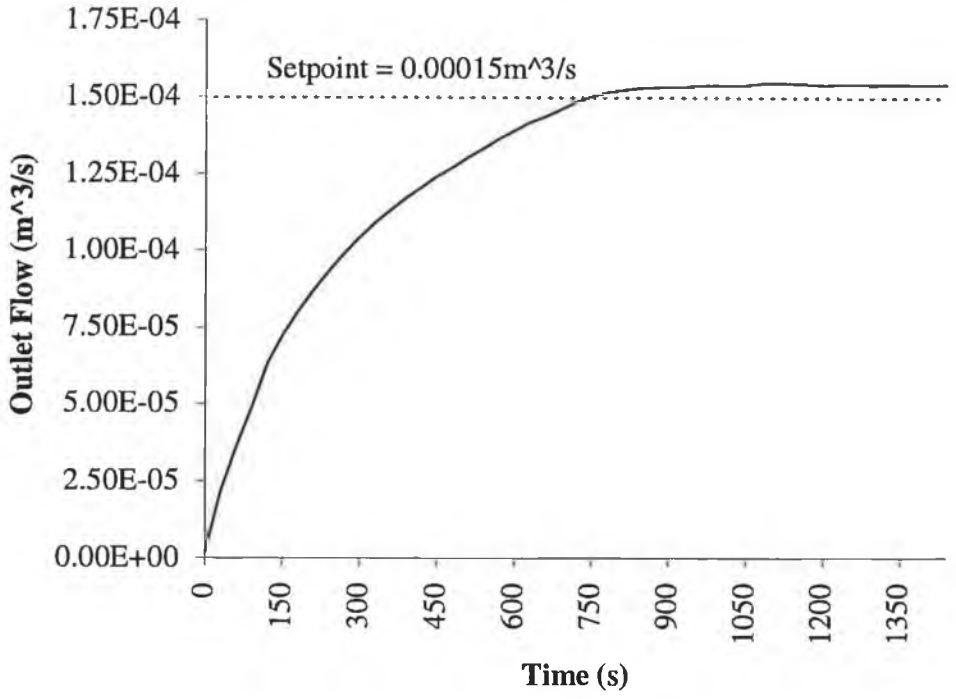


Figure 6.3 - SPFC outlet flow control - setpoint = 0.0001 m^3/s.



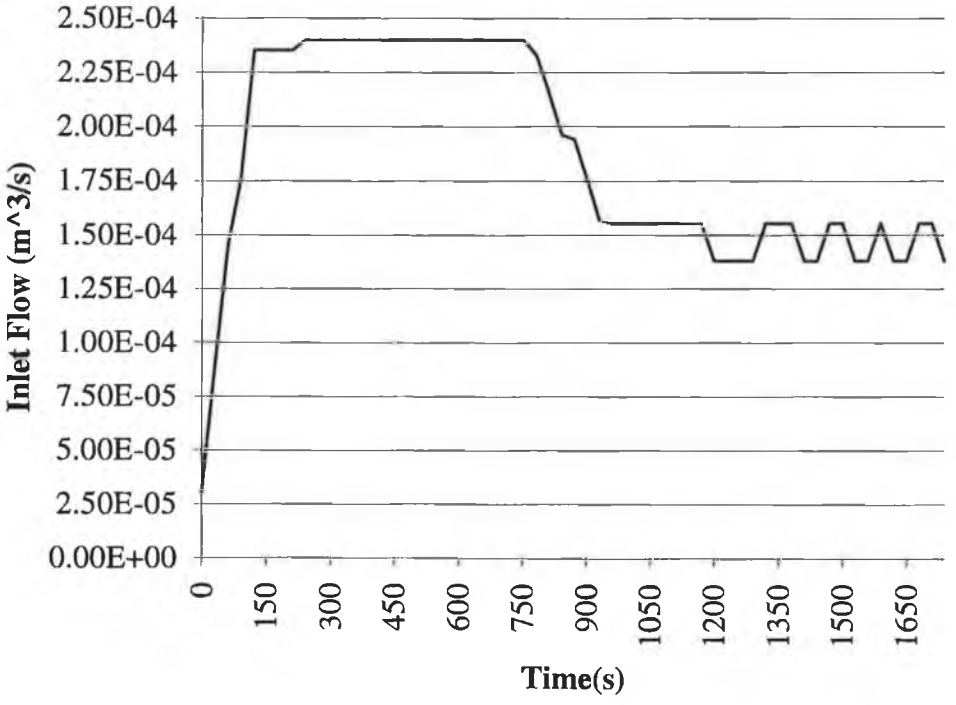
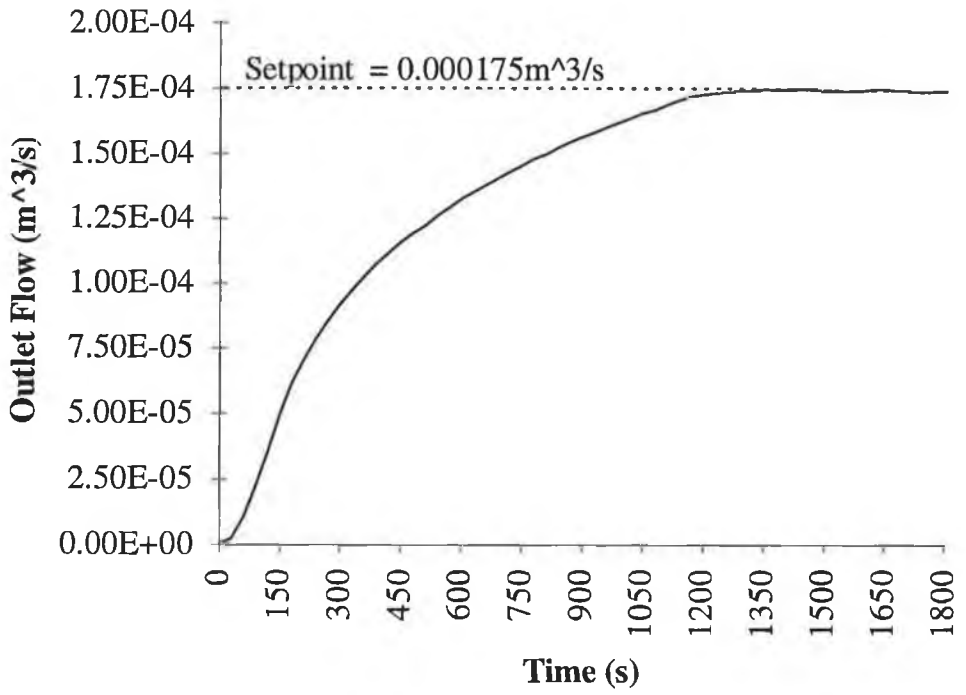


Figure 6.4 - SPFC outlet flow control - setpoint = 0.00015 m^3/s.



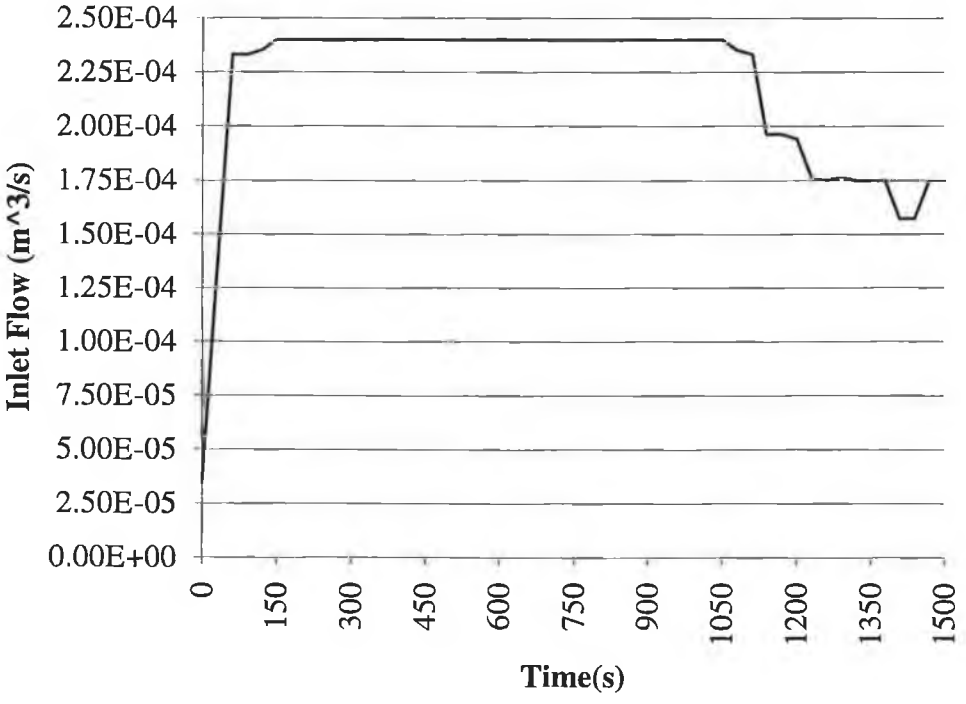
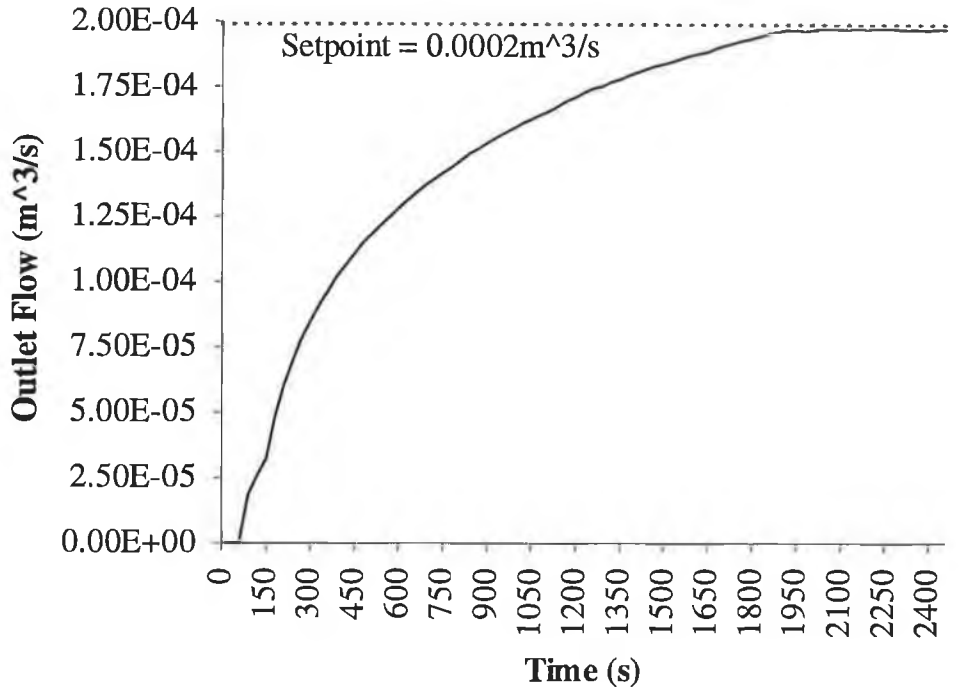


Figure 6.5 - SPFC outlet flow control - setpoint = 0.000175 m^3/s.



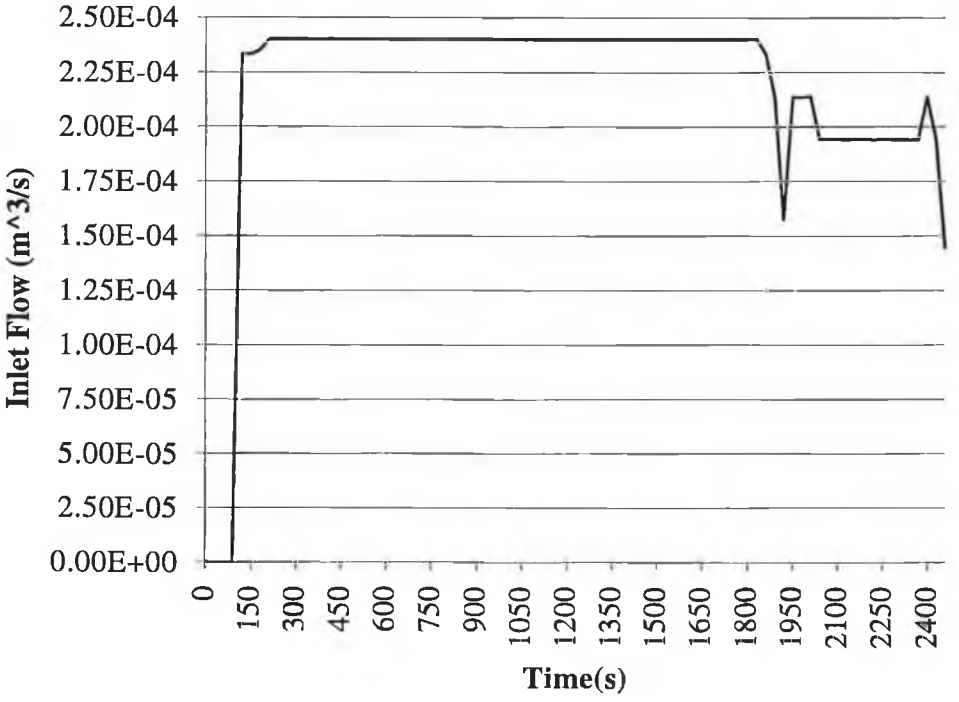


Figure 6.6 - SPFC outlet flow control - setpoint = 0.0002 m^3/s.

The results of these tests show that, in principle, the SPFC is capable of controlling the outlet flow of the real warm water process. Table 6.1 lists the time constants and steady state errors as a percentage of full scale for the six tests.

Table 6.1 - Time constants and steady state error of outlet flow control tests.

Setpoint (ml/s)	*Approximate Time Constant (s)*	*Steady State Error (% of full scale)*
50	110	2.91
75	110	2.08
100	140	2.91
150	270	1.56
175	400	0.8
200	570	1.33

There are three types of controller errors evident in the experimental results :

- *Oscillation at low setpoint values* - caused by the fast plant dynamics at lower outlet flow operating points. The observed oscillation at a flow operating point of 50 ml/s has a period of 180 seconds, whereby the sampling period of 30 seconds is only one sixth of the oscillation period. An increase in the sampling rate to 15 seconds would reduce this oscillation.

- *Inaccurate dynamic responses* - refers to the differences observed between the time constants of the outlet flow responses and the desired time constant of the reference model. These differences are caused by the physical hard limit for both cold and hot inlet flows, the sum of which was 150 ml/s. The desired dynamic response of the first order reference model with a time constant of 120 seconds is not attainable for operating points corresponding to outlet flows larger than approximately 50 ml/s.

- *Steady state errors* - small steady state errors are evident in all outlet flow responses and are due to the *fuzziness* of the fuzzy model. The fuzzy model used in this controller to model the mass flow of the warm water process, uses 21 fuzzy sets for each of the two antecedent variables and a value of $\delta = 0.7$, see the learning equation (4.25). Thus the length of the adaptable areas of each rulebase cell correspond to 3.41% of the full scale value of the input variables. The steady state errors observed during these tests all correspond to values of less than 3% of the full scale value of the outlet flow. Thus, it can be concluded that the controller steady state error is caused by the intrinsic error of the fuzzy model.

To show the adaptation of the rulebase used in the fuzzy model of the SPFC for outlet flow control, the percentage change in the rulebase cells with respect to their initial values is shown in Figure 6.7. By using the MATLAB command *spy()*, the number of adapted rulebase cells was seen to be 229 which corresponds to 62% of the rulebase.

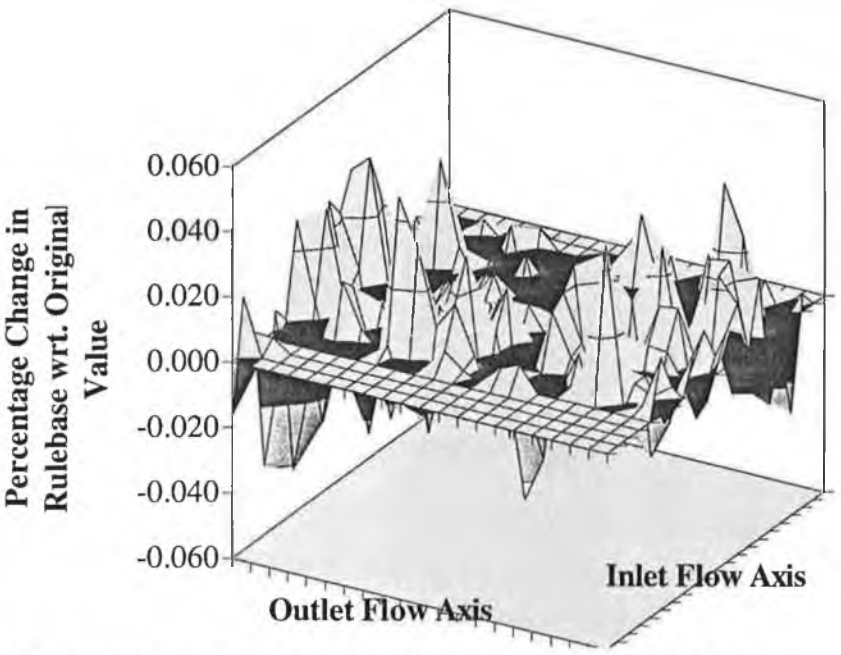


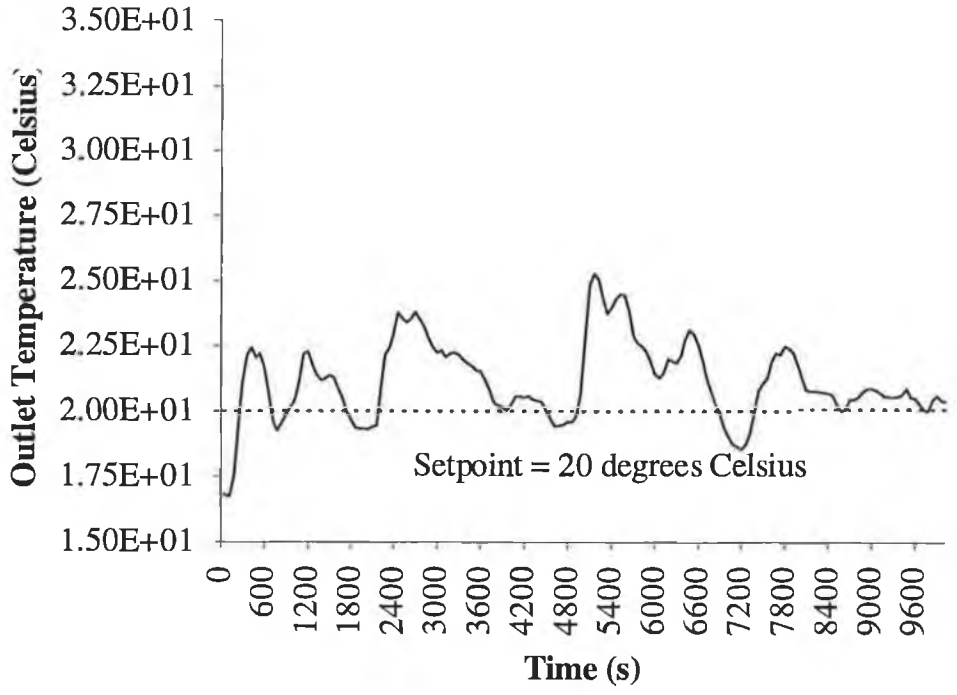
Figure 6.7 - Ratio of change in rulebase cells with respect to initial values.

The changes in the rulebase are all less than 0.1 percent of the initial values, thus the initial rulebase can be assumed to be an accurate representation of the mass flow of the warm water process.

6.3. Warm Water Process Outlet Temperature Control

A *set of five experiments* were carried out on the plant for MISO control of the outlet temperature with setpoints of 20, 22.5 ,25, 27.5 and 30 degrees Celsius. The results of these five tests are shown in Figures 6.8, 6.9, 6.10, 6.11 and 6.12 together with the hot inlet flow temperature disturbance variable. In order to increase the speed of response of the plant, the first order reference model used in the SPFC for these tests, was given a time constant of 300 seconds.

The *hot inlet temperature* is a measurable disturbance variable, whereby its value is dependent on the use of hot water in the building and the mode of operation of the central boiler used for heating the water. No influence on either the consumption of hot water or the mode of operation of the boiler was possible during the course of the experiments.



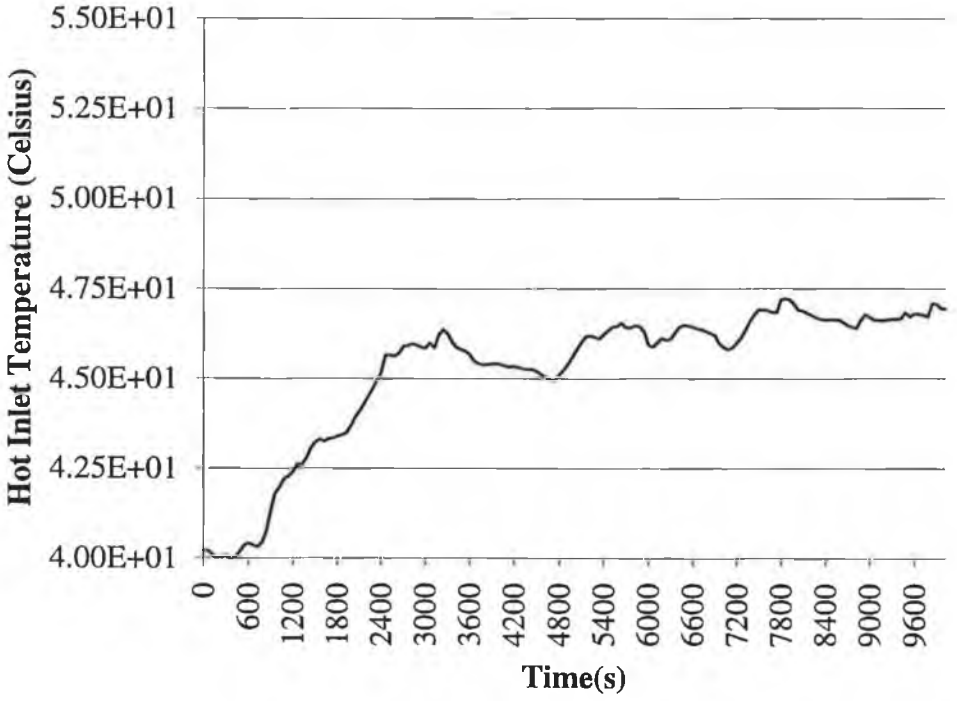
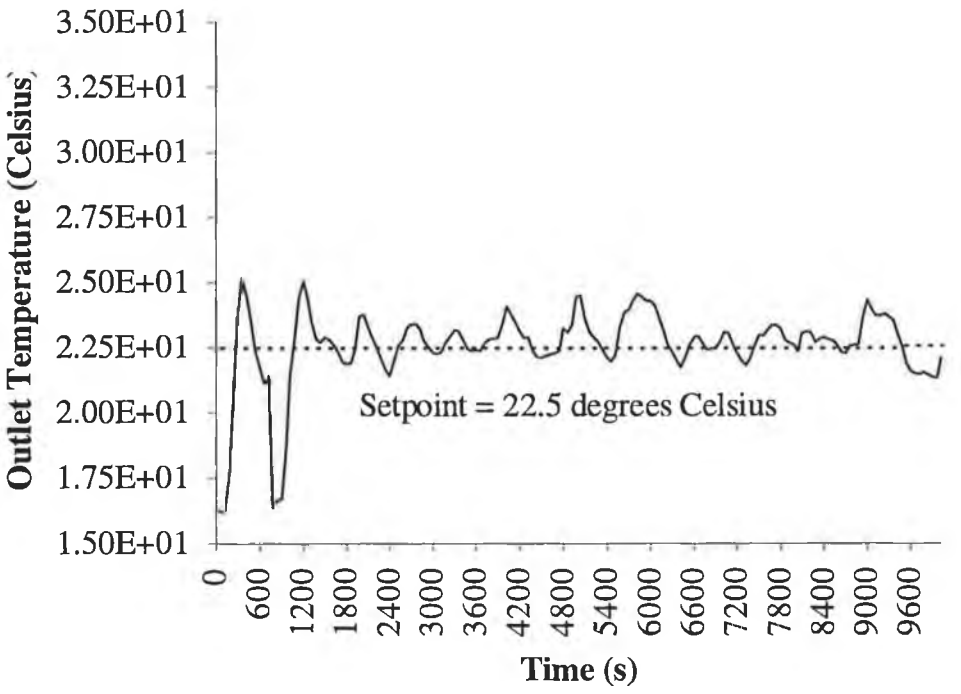


Figure 6.8 - SPFC outlet temperature control - setpoint = 20 degrees Celsius.



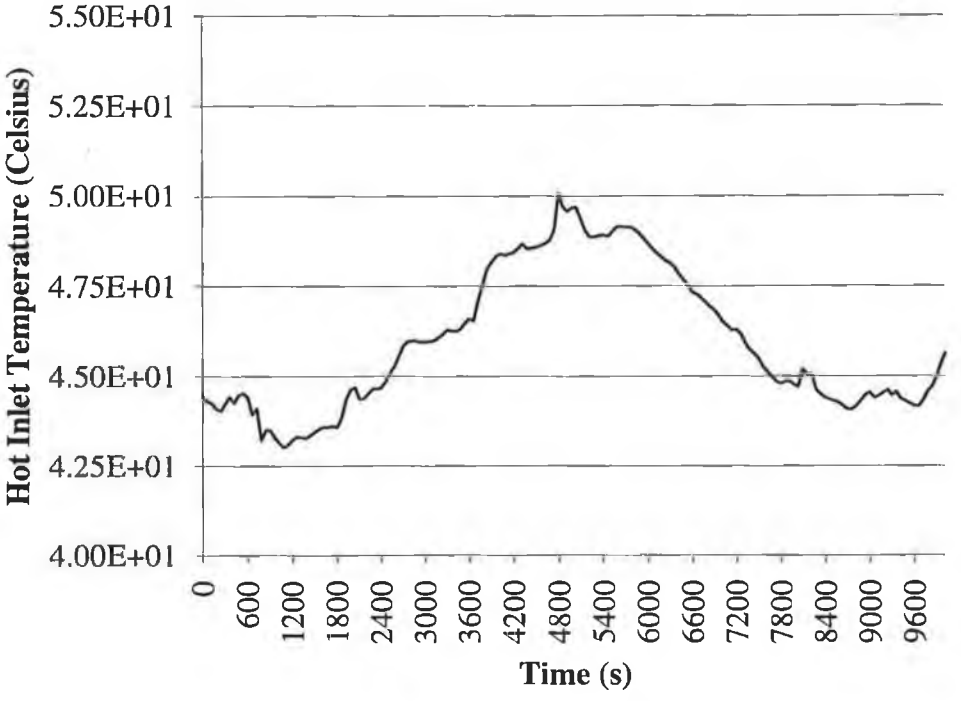
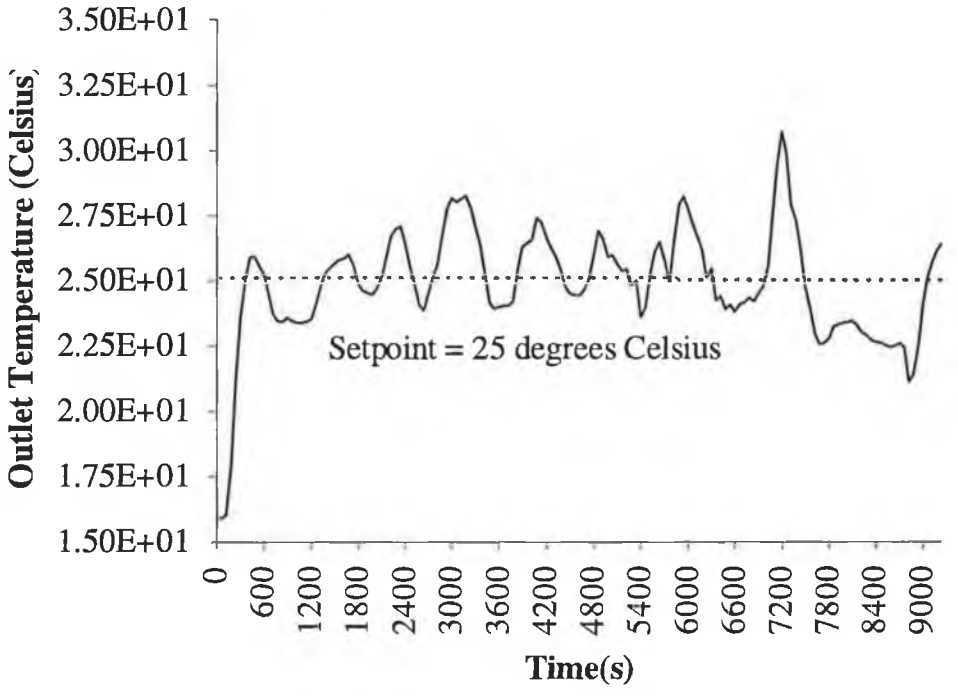


Figure 6.9 - SPFC outlet temperature control - setpoint = 22.5 degrees Celsius.



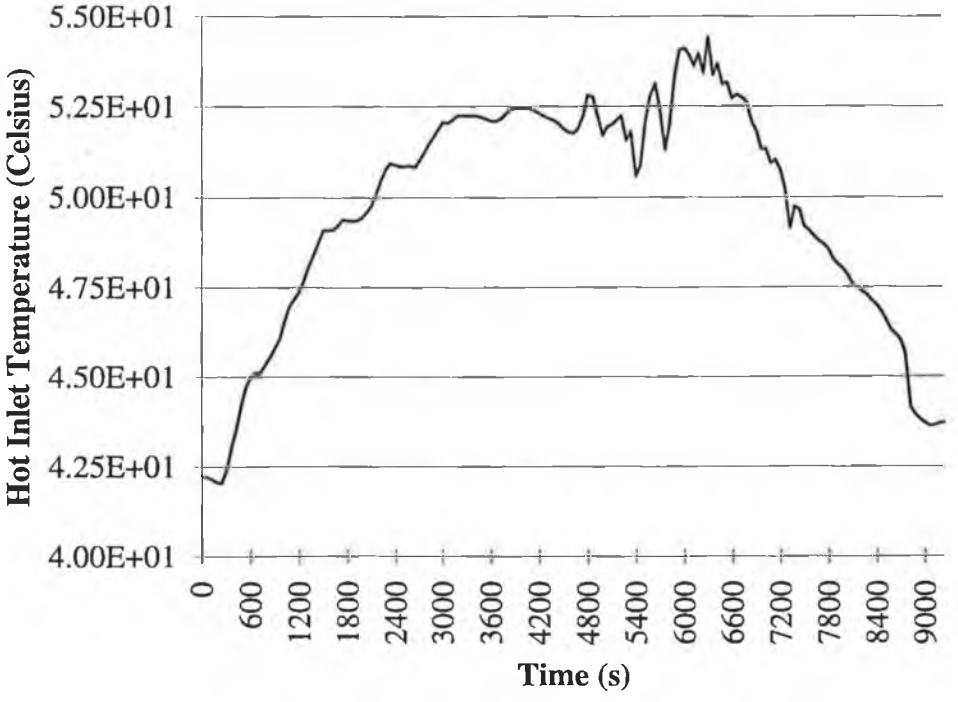


Figure 6.10 - SPFC outlet temperature control - setpoint = 25 degrees Celsius.

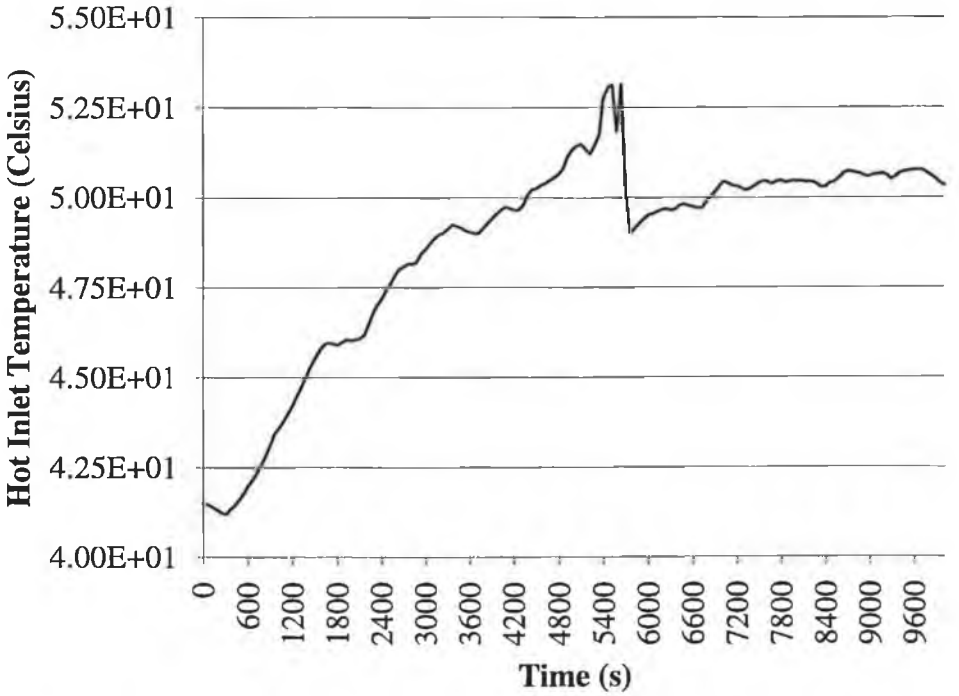
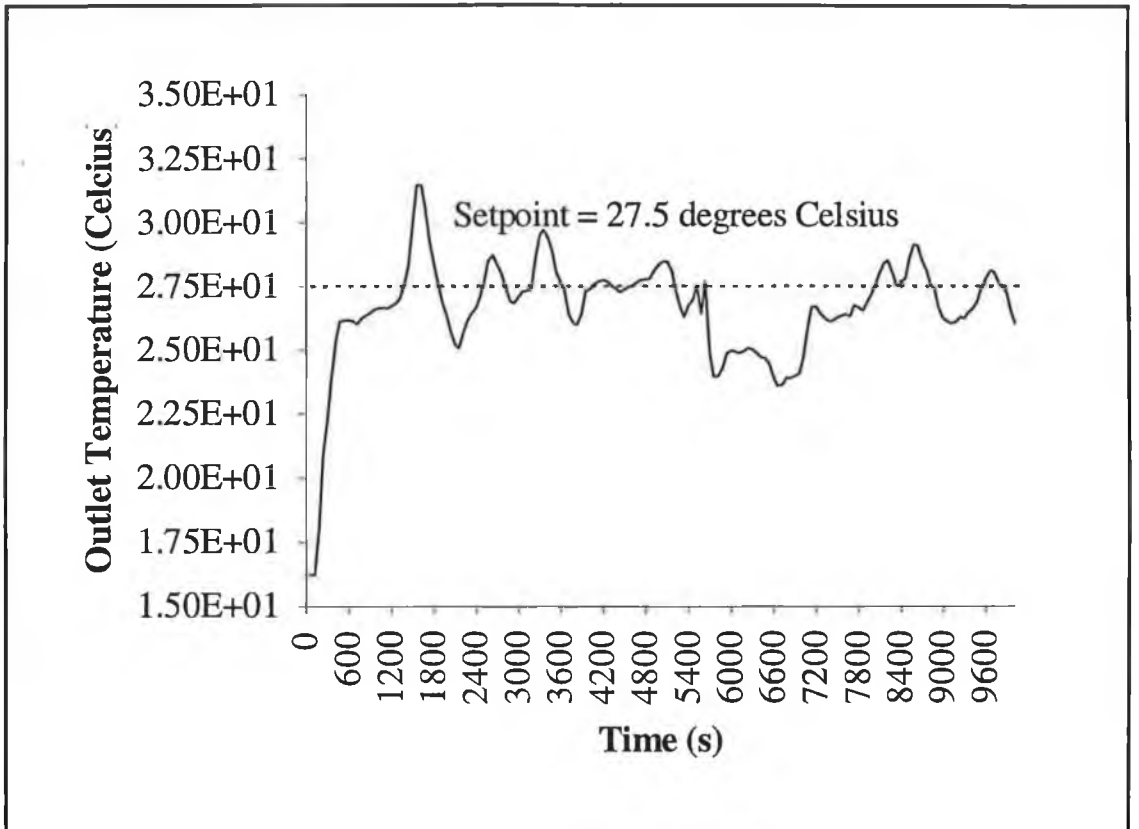


Figure 6.11 - SPFC outlet temperature control - setpoint = 27.5 degrees Celsius.


Outlet Temperature (Celcius
3.50E+01
3.25E+01
3.00E+01
2.75E+01
2.50E+01
2.25E+01
2.00E+01
1.75E+01
1.50E+01
Setpoint = 27.5 degrees Celsius
0
600
1200
1800
2400
3000
3600
4200
4800
5400
6000
6600
7200
7800
8400
9000
9600
Time (s)

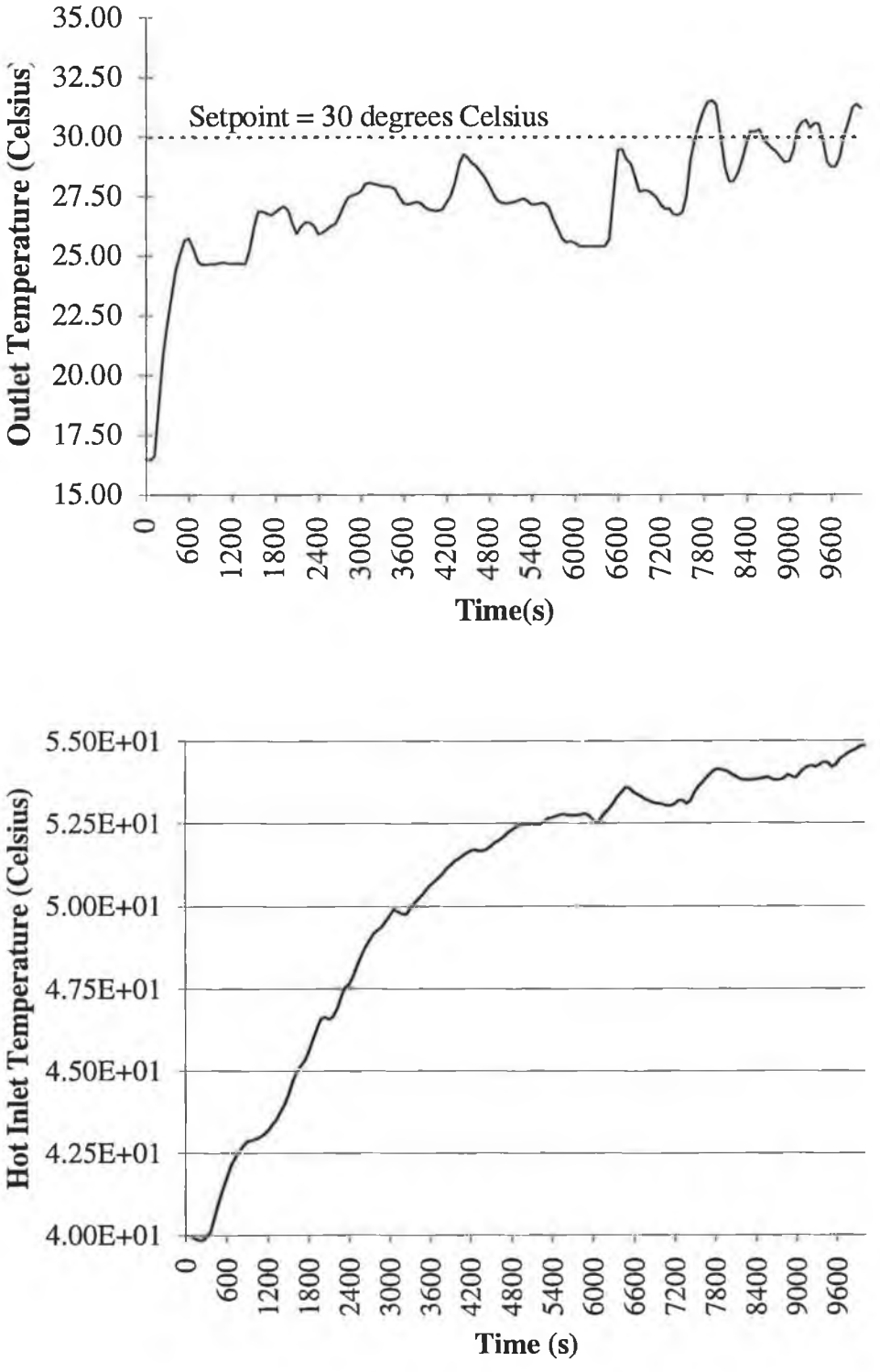


Figure 6.12 - SPFC outlet temperature control - setpoint = 30 degrees Celsius.

The SPFC used for *outlet temperature control* searches for the optimum value of both the cold and hot inlet flow values. Moreover, the rulebase to be searched has four dimensions and requires more processing time per search iteration compared to the two dimensional rulebase of the outlet flow SPFC. Due to the increased processing time requirements, combined with the relatively slow Intel 386 processor and the 20Hz interrupt load (from the ISR for inlet valve control), the sampling time of 30 seconds was not sufficient to allow a full search of the rulebase. The sampling time was thus increased to 60 seconds, whereby this represents one tenth of the thermal time constant derived in Section 4.4 and is still sufficient to capture the thermal dynamics of the plant.

As seen from Figures 6.8 to 6.12, the responses obtained in these experiments are of an oscillatory nature. In order to evaluate these responses, the mean and standard deviation of the outlet temperature *after* rise time was calculated, see Table 6.2.

Table 6.2 - Mean and standard deviations of the outlet temperature tests.

Setpoint (° Celsius)	***Mean (° Celsius)***	***Standard Deviation (° Celsius)***
20	21.2	1.43
22.5	22.7	1.20
25	25.1	1.71
27.5	26.8	1.51
30	27.5	2.75

The *oscillation and steady state errors* (based on the mean outlet temperature values after rise time) evident in all of the outlet temperature control experiments described in this Section are due to the following (in order of effect) :

- *Poor disturbance rejection of the SPFC outlet temperature controller* - which is discussed in Section 5.3.3. of this thesis. One example of the effect of the hot inlet temperature on the SPFC response can be observed for the setpoint of 20 degrees Celsius (see Figure 6.8). The peaks in the hot inlet temperature at 3000, 5400, 6600 and 8000 seconds cause corresponding oscillations at the same times in the controlled variable.

- *Limited prediction horizon of the SPFC* - does not allow sufficient compensation of the mixing dynamics of the plant.

- *Fast reference dynamic* - which causes large variance in the manipulated variables.

- *Fuzzy model errors* - in the fuzzy model of the thermal behaviour of the warm water process that was used in the SPFC. As described in Section 4.9.3 each of the antecedent variables used in this model has 15 fuzzy membership sets. Thus the maximum resolution of this fuzzy model is 1.1 degrees Celsius (assuming an outlet temperature span of 40 degrees Celsius).

The adaptation of the rulebase was investigated using the spy() command in MATLAB. Approximately 25.6% of the rulebase cells were adapted whereby the average percentage change of the adapted rulebase cells with respect to their original values was approximately 15%. This implies that the initial fuzzy model of the warm water process was reasonably accurate.

6.4. Conclusions

The control of the outlet flow of the warm water process corresponded well with the simulation results described in Chapter 5. The results obtained for outlet temperature control of the warm water process using the SPFC were of poor quality.

To *improve the quality of control of the SPFC* for the control of the outlet flow and temperature of the warm water process, the *following enhancements* are necessary :

- *Increased fuzzy model accuracy* - for the fuzzy models used for the prediction of the outlet flow and temperature in the SPFC through an increase in the number of cells in the respective rulebases. For the fuzzy model of the thermal behaviour of the warm water process, the replacement of the hot enthalpy input variable with separate hot inlet flow and temperature variables would improve the disturbance rejection of the SPFC for temperature control. As described in Section 5.3.3 this would require the use of another operating system as the memory capacity of DOS would not be sufficient.

- *Extension of the prediction horizon* of the SPFC, thus forming the Multi-Step Predictive Fuzzy Controller (MPFC). The extension of the prediction horizon would allow improved compensation of the non-linearities of the warm water process, especially in the case of the thermal mixing dynamics. Chapter 7 of this thesis presents a concept for the development of the MPFC.

- *Slower reference model dynamics* - would reduce the forward loop gain of the SPFC, better controller performance would be achieved for the control of the

outlet temperature, if the time constant of the first order reference model were considerably larger than the 210 seconds used in these experiments e.g. 600 seconds.

Unfortunately, due to the access restrictions and compressor breakdowns mentioned at the beginning of this chapter, the suggested improvements for the SPFCs for the outlet flow and temperature control could not be performed.

Chapter 7 - *Conclusions and Suggestions for Further Research*

7.1. Introduction

This chapter draws conclusions concerning adaptive fuzzy control from the research undertaken in this thesis and suggests areas where further research could be performed. *Section 7.2* of this chapter contains suggestions for further research and *Section 7.3* draws overall conclusions from the research presented in this thesis.

7.2. Suggestions for Further Research

7.2.1. Modelling of the Warm Water Process

The *physical model of the warm water process* developed in this research proved to be a sufficient model of the mass flow but was unable to model the *non-linear thermal behaviour* of the plant adequately. The addition of extra sensors along the length of the process reaction tank of the warm water process would, by means of data acquisition, enable more exact physical models of the thermal behaviour of the warm water process to be developed.

Although not directly concerned with the topic of this research, the *ANN model of the warm water process* played a key role in *initialising* the fuzzy models for the on-line control of the warm water process. Moreover, the ANN model was used as a simulation model of the warm water process for some of the simulation work performed in Chapter 5. Based on the behaviour of the ANN warm water process model observed during the course of these simulations, it became clear that the training data used to train the ANN model did not globally represent the state space of the warm water process adequately.

As described in Chapter 1, one aspect of the area of neural-fuzzy research utilises fuzzy algorithm representations of human expert knowledge to initialise neural networks, which are then further trained with measured system data. In order to improve on *the global modelling capability* of ANN model of the warm water process and to improve its *training time*, initialisation of the ANN with data taken from either the physical model or a human expert could be performed. This would initialise the network with a state more representative of the process dynamics than the set of small random weight values normally used and thus could help to shorten the training times. Moreover, this initialisation could help to fill any gaps in the state space not provided by measured system data and thus improve the global nature of the model. Further modelling using such ANN architectures as radial basis networks

would also be of interest, as these offer faster training times without the paralysis and local minima problems of the backpropagation trained ANN but at the expense of on-line computation time [101].

The *fuzzy models* utilised in this research possess a rulebase in lookup table format. In order to increase the accuracy of these fuzzy models, *relational matrix rulebases* could be used. Research by Küpper [108] has suggested a method of training a fuzzy model with a relational matrix rulebase using a fast and convergent stochastic learning algorithm. In order to improve model convergence, this paradigm could be augmented by the supervisory function suggested in Chapter 4 of this research.

7.2.2. Self-Organising Controller (SOC)

As detailed in Chapter 2 of this thesis, the most prominent class of stand-alone adaptive fuzzy controller is the direct strategy of the *Self-Organising Controller (SOC)* where the rule consequents of a rulebase are adapted. This thesis shows that the SOC is a *complex and computationally intensive* algorithm which has relatively *poor control performance*. One of the main problems with this controller strategy is the large number of parameters for which no direct relationship to controller response can be perceived. Of these parameters, the *reference model* has the most direct influence on the SOCs control performance. However, in all of the literature found, the reference model is of a heuristic nature, often in the form of a lookup table with up to 200 values, and has not been concisely investigated. One possible method *to reduce the number of SOC parameters while increasing the clarity of the SOC design* would be to replace the heuristic reference model with a deterministic algorithm. This modification would allow the user to directly specify the desired response of the controller, thus making a simpler specification and evaluation of its performance possible. If a first order system, for example, were used as a reference model, then only the steady state gain and time constant would need to be specified, both of which have clear physical interpretations and allow definitive evaluations of the SOC performance.

Associated with the utilisation of a direct and transparent deterministic reference model, more attention should be given to the determination of *the causal relationship between the consequent values of past rulebases and the current controller performance*, which decides precisely which elements of the current rulebase are to be adapted. In practice a delay (most often a single parameter m) in rule consequent adaptation is specified, based on the fact that past controller rulebases have contributed to current controller performance. Different publications have claimed

that the exact determination of this relationship (and thus the value of this delay parameter) is unimportant, as the SOC is able to compensate for its false specification. This would, however, seem unlikely and needs to be rigorously investigated.

The issue of *adaptation of the rulebase* is a crucial factor in the SOC design. In most of the SOC implementations, the adaptation is largely dependent on the position of the controlled variable in the state space, even though the controlled variable may be moving towards the setpoint in a satisfactory manner. If adaptation were made dependent on the error signal trajectory, then rulebase adaptation would only be performed if the trajectory of the controlled variable is unsatisfactory, and would provide improved SOC performance. Such trajectory based adaptation could be achieved by comparing the trajectories of a deterministic reference model and the controlled variable, basing the adaptation on the difference between the two. Due to the interpolative and inexact nature of the fuzzy controller, adaptation based on small controller errors is not meaningful and can lead to instability of the SOC algorithm. Thus, some research should be directed at attempting to define refined criteria for adaptation.

7.2.3. Fuzzy Modelling

Fuzzy modelling plays a key role in this research as it is an integral part of the single step predictive fuzzy controller strategy which is applied to the control of the outlet flow and temperature variables of the warm water process. The key advantage of the *supervised adaptive fuzzy modelling strategy* developed in Section 4.9 of this thesis is its ability to learn non-linear mappings on-line using a simple adaptation algorithm. The main disadvantage of this modelling strategy is its memory intensive nature, which increases exponentially with the number of model inputs.

Because the performance of this indirect adaptive fuzzy controller strategy is directly dependent on the accuracy of the fuzzy model used, more research is required into improving the modelling accuracy and memory requirements of *on-line* adaptive fuzzy models. Such research should include :

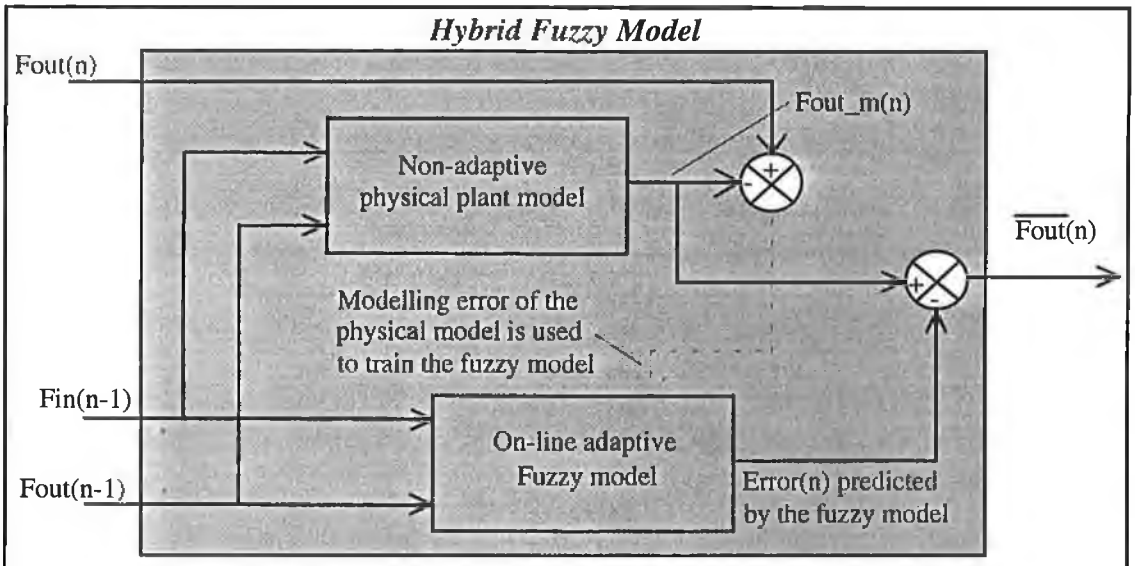
- the determination of *realistic trade-offs* between memory requirements and modelling accuracy,

- the investigation into the utilisation of *different learning algorithms*,

- the development of *structurally adaptive models* which would start with small rulebases and, as the rulebase fills, would automatically increase rulebase size and resolution and

- the combination of fuzzy models and physical models to form *hybrid fuzzy models*.

Figure 7.1 shows one example of the *synthesis of a fuzzy model and a deterministic model* using the mass flow of the warm water process as an example. Here, the on-line adaptive fuzzy model is used to learn the modelling error of the physical plant model and, through compensation, improve overall modelling performance.



Fin(n-1) is the previous inlet flow of the real warm water process.
Fout(n-1) is the previous outlet flow of the real warm water process.
Fout(n) is the current outlet flow of the real warm water process.
Fout_m(n) is the predicted outlet flow from the physical model.
$\overline{\text{Fout}}(n)$ is the predicted total outlet flow of the warm water process.
Error(n) is the modelling error of the physical model predicted by the fuzzy model.

Figure 7.1 - Synthesis of adaptive fuzzy model and physical plant model.

Such *hybrid fuzzy models* would offer improved modelling performance as the fuzzy model would no longer be required to learn the complete system behaviour. The know-how contained in physical models can be utilised whereby the fuzzy model would serve to increase its accuracy. Furthermore, in contrast to the fuzzy models used in this research, the initialisation of the fuzzy model would not be necessary, as the physical model would provide the initial output of the overall hybrid model.

7.2.4. Predictive Fuzzy Control

This research used a simple *single step predictive fuzzy controller* (SPFC) as a basis and enhanced it through the extension of the step reference model to a first order system, thus allowing better definition of its dynamic response. Through application to a non-linear multivariable plant, the advantages and shortcomings of the SPFC were highlighted. The main limitation of the SPFC was the single step prediction horizon which was not sufficient for the prediction of non-linear plant dynamics over enough of a time period, to enable the control of systems with strongly non-linear dynamics to a reasonable degree of accuracy.

As a solution to the limited prediction horizon, Chapter 7 develops a method for the extension of the SPFC to the *multi-step predictive fuzzy controller* (MPFC). This is achieved through the utilisation of differentiable fuzzy models and the application of a gradient descent algorithm to the proposed MPFC structure to optimise the values of the manipulated variables. This MPFC could serve as the focus for further research, where the main issues are deemed to be:

- investigation of the achievable *speed and quality of optimisation* through the utilisation of a variety of gradient descent methods,
- the determination of the effect of the *accuracy of fuzzy models* on the controller performance,
- comparison and contrast with *classical predictive control* strategies and
- the use of *hybrid fuzzy models*, as proposed in Section 7.2.3 of this chapter, in a multi-step predictive controller.

7.3. Conclusions

Fuzzy control offers the engineer a method with which it is *possible* to control a plant using *human expert knowledge stored in a mathematical form*. This capability has achieved *some success* in the control of ill-defined systems where classical control theory had failed to provide reliable automated closed loop control. However, fuzzy control does not offer a general solution for all difficult control problems as implied by some of its more aggressive proponents. Precisely these exaggerations coupled with the intrinsic ad-hoc nature and poor quality of much of the literature concerning fuzzy control have lead to acceptance problems among the closed loop control community.

A fuzzy controller performs nothing other than an *interpolation between the rules in its rulebase* and can be viewed as an *interpolating lookup table*, where the elements in the lookup table can be referenced by linguistic terms. The form of interpolation used is dependent on such parameters as the type and number of fuzzy membership set functions, the inference and aggregation functions and the defuzzification method. The major disadvantage with the fuzzy controller is the selection and optimisation of the large number of parameters. The designer can reduce the number of these parameters by reducing the size of the rulebase, but while this solution offers higher clarity of design, the overall accuracy of the fuzzy controller will suffer.

Fuzzy controllers are *not suitable* for high accuracy control tasks in such applications as robotics. This unsuitability is due to the interpolative nature of the fuzzy logic controller and the difficulty in relating controller parameters to controller performance when a fuzzy controller is utilised. Such control tasks are better left to classical control methods which, by means of their deterministic nature, provide higher accuracy and often allow optimisation of the controller for a given cost function around a given plant operating point. Only in the case where classical control algorithms fail to deliver the desired controller response, should the implementation of a fuzzy controller be attempted.

The field of adaptive fuzzy control was categorised into three areas in this research with most attention being given to stand-alone adaptive fuzzy control strategies. The majority of adaptive fuzzy control techniques found are of *limited practical value*. As far as the author is aware, the only type of adaptive fuzzy controller that has proven to be commercially viable, is the self-tuning PID controller with a fuzzy tuner which is available from the company OMRON [1].

The *single step predictive fuzzy controller* chosen for practical evaluation in this research is of limited practical value due to the following disadvantages :

- *single step prediction* is not sufficient for good control of non-linear multivariable plants.

- *extravagant memory requirements* - the fuzzy model used for thermal behaviour modelling requires over 200 kBytes of memory.

- *excessive computational requirements* with of the controlled search algorithm used - one calculation of the output of the four dimensional fuzzy

model of thermal behaviour required approximately 0.05 seconds when run on a 386DX 33Mhz PC.

- the *quality of control* is overly-dependent on the modelling accuracy of the fuzzy model used.

The development of the *multistep predictive fuzzy controller*, suggested in Chapter 7, coupled with the continual decrease in the price of computing equipment and their steadily improving performance may make the predictive fuzzy controller interesting for practical implementations in the future, but for the moment it will remain largely within the realms of academic research.

In most of the literature found, the authors attempt to develop adaptive fuzzy control strategies without the incorporation of know-how from classical control theory. This results in ad-hoc control paradigms such as the self-organising controller and many of the gain tuning/adaptation methods found. It is beyond doubt that adaptive fuzzy control can only serve to gain by exploiting the knowledge and experience available in the field of classical control. The synthesis of adaptive fuzzy control and classical control theory would benefit control theory as a whole, through the exploitation of the advantages of both methods, to overcome their individual weaknesses.

Appendix A

Table A.1 - List of components used in signal conditioning board.

Resistors	***Value***	***Tolerance***
R1	470 Ω	5%
R2	2.2 MΩ	5%
R3	2.2 MΩ	5%
R4	1 MΩ	5%
R5	1 MΩ	5%
R6	***See Table A.2***	5%
R7	1 MΩ	5%
R8	1 MΩ	5%
R9	1.8 MΩ	5%
R10	2.2 MΩ	5%
R11	470 kΩ	5%
R12	680 kΩ	5%
Variable Resistors **- Carbon types**		
RV1	100 Ω	10%
RV2	1 MΩ	10%
RV3	47 kΩ	10%
Capacitors		
C1	2.2 nF Ceramic	10%
C2	***See Table A.2***	10%
C3	2.2 nF Ceramic	10%

Semiconductor Devices

A1 and A2 are 2 operational amplifiers in a quad-opamp integrated circuit - TL074 from Texas Instruments.

Table A2 - Anti-aliasing cut off frequencies and corresponding component values.

Plant Variable	***Cut-off Frequency (Hz)***	***Value of C2 in signal conditioning board channel***	***Value of C2 in signal conditioning board channel.***
Cold Inlet Flow	5	22 µF	47 kΩ
Hot Inlet Flow	5	22 µF	47 kΩ
Process Reaction Tank Level	0.0833	22 nF	47 kΩ
Outlet Flow	0.0833	22 nF	47 kΩ
Hot Inlet Temperature	0.0833	22 nF	47 kΩ

Outlet Temperature	0.0833	22 nF	47 kΩ

Table A.3 - List of components used in the two power supplies.

Capacitors	
C1, C2, C3	1000μF
Transformers	
T1	24VA 0-220/240V primary voltage, 1x24V secondary voltage at 1 ampere, chassis mount
T2	30VA 0-220/240V primary voltage, 2x15V secondary voltage at 1 ampere, chassis mount
Semiconductor Devices	
IC1	MC7824CT Voltage regulator - +24 volts
IC2	MC7815CT Voltage regulator - +15 volts
IC3	MC7915CT Voltage regulator - -15 volts
B1, B2	General Instrument GBPC1 Bridge rectifier

Appendix B

This appendix contains the source listings for the functions necessary for the installation and de-installation of a user defined interrupt service routine. The name of this ISR is ***adc_isr***.

The following ANSI C function installs a interrupt service routine for the IRQ x.

```
/*****************************************************************************
** Program  : INT_INIT.C
** Revision : 1
** Date     : 10.05.93                    STEPHEN MCCORMAC
*****************************************************************************
This file initialises the interupt structure required in this programme.
The ADC card sends an interupt on IRQ-x when ADC is complete.  The module
ADC_ISR.C is the Interupt Service Routine for this IRQ.

*****************************************************************************
****************  included files  ************************************/
#include <dos.h>
#include <stdio.h>
#include <conio.h>
/*****************************************************************************/

/*****************************************************************************
****************  function prototypes  ******************************/


extern unsigned char PICOld;
extern void interrupt AdcIsr(void);
extern void interrupt (*OldVect)();

/*****************************************************************************/

void IntInit(unsigned int IntNum, unsigned int IntEn)
{

/*----------------- declarations-------------------------------------*/
unsigned char TempByte0;
/*----------------- start of main instructions ----------------------*/
disable();                          /* disables all interupts */

TempByte0=inportb(0x21);            /* read in Interupt Flag Register */

PICOld=TempByte0;                   /* save old IMR value */

TempByte0=TempByte0 & IntEn;        /* unmask IRQx */

outportb(0x21,TempByte0);           /* enable IRQ7 interupt */

OldVect=getvect(IntNum);            /* get BIOS installed IRQx interupt
                                       vector */

setvect(IntNum,AdcIsr);                     /* set IRQxvector to adc_isr */

enable();                           /* enable interupts */
```

```
}
```

The following ANSI C function deinstalls the previously installed ISR and restores the PC to its previous state.

```
/*****************************************************************************
** Program  : terminat.c                                                    *
** Revision : 1                                                             *
** Date     : 25.05.93                    STEPHEN MCCORMAC                 *
*****************************************************************************
This module allows the proper termination of the control system for the warm
water process.
*****************************************************************************
***************  included files  ************************************/
#include "par_sys.h"
#include <dos.h>
/*****************************************************************************/

extern unsigned char PICOld;
extern void interrupt AdcIsr(void);
extern void interrupt (*OldVect)();

/*****************************************************************************
***************  function prototypes  ********************************/
extern void DacInit(void);
/*****************************************************************************/

void Terminate(unsigned int IntNum)
{
/*---------------- declarations-------------------------------------*/

/*---------------- start of main instructions ----------------------*/

DacInit();                          /*sets all DAC outputs to zero */

outportb(AdcIntEn,0xff);            /* reset interrupt request on ADC */
outportb(AdcModeCont,0x0);        /* reset ADC card trigger modes */

outportb(0x20,0x20);              /* verify interrupt */

disable();                          /* disables all interupts */

outportb(0x21,PICOld);              /* restore old PIC value */

setvect(IntNum,OldVect);            /* restores old interrupt vector */

enable();                           /* enable interrupts */

}
```

The following ANSI C function multiplies two matrices together.

```
#include <stdio.h>
#include <structs1.h>

extern MATRIX matinit1(unsigned int ROW, unsigned int COL, MATRIX  huge
*Mat);

MATRIX MULM(MATRIX huge * A,MATRIX huge * B)
```

```
{
MATRIX ANS;
int i=0,j=0,k=0,err=0;
float sum=0.0;


/* check matrix dimensions */
if(A->columns!=B->rows)
  {
  printf("\n\a Mismatching matrix dimensions in MULM");
  err=1;
  }

/* initialise answer matrix */
ANS=matinit1(A->rows,B->columns,&ANS);

if(err==0)
{
for(i=0;i<ANS.rows;i++)
  for(j=0;j<ANS.columns;j++)
     {
     sum=0.0;
     for(k=0;k<A->columns;k++)
         sum+= A->mat[i][k] * B->mat[k][j];
     ANS.mat[i][j]=sum;
     }
}



return(ANS);
}
```

The following ANSI C function represents the artificial neural network model of the warm water process.

```
/*****************************************************************************

This function calculates the output of  the ann model of
the icc tank.

*****************************************************************************/

#include <structs1.h>

extern MATRIX   DIVV(MATRIX huge *,MATRIX huge *);
extern MATRIX        MULV(MATRIX huge *,MATRIX huge *);
extern MATRIX        MULM(MATRIX huge *,MATRIX huge *);
extern MATRIX        LOGSIGV(MATRIX huge *,MATRIX huge *);
extern MATRIX        ADDM(MATRIX huge *,MATRIX huge *);
extern void     matfree1(MATRIX huge *);

MATRIX nnrun(MATRIX huge * u,MATRIX huge * w1m,MATRIX huge *
w2m,MATRIX huge * w3m,
          MATRIX huge * b1m,MATRIX huge * b2m,MATRIX huge * b3m,
          MATRIX huge * pscalem,MATRIX huge * tscalem)
{
MATRIX u1,u2,u3,u4,u5,u6,u7,y;
```

```
u1=DIVV(u,pscalem);
u2=MULM(w1m,&u1);
u3=LOGSIGV(&u2,b1m);
u4=MULM(w2m,&u3);
u5=LOGSIGV(&u4,b2m);
u6=MULM(w3m,&u5);
u7=ADDM(&u6,b3m);

y=MULV(&u7,tscalem);

matfree1(&u1);
matfree1(&u2);
matfree1(&u3);
matfree1(&u4);
matfree1(&u5);
matfree1(&u6);
matfree1(&u7);

return(y);

}
```

The following ANSI C source code listing allocates memory for a four dimensional rulebase.

```
/* function to initialise a matrix of doubles*/
#include <stdio.h>
#include <structs1.h>
#include <stdlib.h>
#include <alloc.h>
#include <conio.h>

extern MATRIX        matinit1(unsigned int, unsigned int, MATRIX  huge *);


RB4D rb4dinit(unsigned int ROW_2,unsigned int COL_2,
          unsigned int ROW, unsigned int COL, RB4D huge *Mat)

{
unsigned int i=0,j=0;

MATRIX huge ** array;

Mat->rows_2=ROW_2;
Mat->columns_2=COL_2;

array=(MATRIX huge **)calloc(ROW_2,sizeof(MATRIX));

if(array==NULL)
  printf("\n!!!Allocation of RB4D failed.");

for(j=0;j<ROW_2;j++)
  {
  array[j]=(MATRIX huge *)calloc(COL_2,sizeof(MATRIX));
  if(array[j]==NULL)
    printf("\n!!!Allocation for rows failed.");
  }

for(i=0;i<ROW_2;i++)
```

```
  for(j=0;j<COL_2;j++)
    {
    array[i][j]=matinit1(ROW,COL,&array[i][j]);
    }



Mat->matp=array;

return(*Mat);
}
```

The following ANSI C source code listing is of the function used to for rule antecedent inference for four antecedent variables followed by the centre of gravity defuzzification for the corresponding lookup table rulebase with singleton consequents.

```
#include <infrb4d.h>
#include <stdio.h>
#include <structs1.h>
#include <alloc.h>
#include <stdlib.h>

extern float    lookup4d(unsigned int ROW_2,unsigned int COL_2,unsigned int
ROW,unsigned int COL,RB4D huge * MAT);
extern void     printmat(MATRIX huge *);
extern MATRIX          matinit1(unsigned int,unsigned int,MATRIX huge *);
extern void     matfree1(MATRIX huge *);

extern float    INFPROD(MATRIX huge *);
extern float    ALGSUM(MATRIX huge *);
extern float    MEAN(MATRIX huge *);
extern float    MINIMUM(MATRIX  huge *);
extern float   FUZOR(MATRIX huge *, float);
extern float    MAXIMUM(MATRIX huge *);
extern float    FUZUND(MATRIX huge *,float);

int infrb4d(MATRIX huge *crisp, MATRIX Dofs[],MATRIX DofsNZ[], RB4D huge *
RuleBase,int IF,float gamma)
{
register int i,j,k,l,m;
register float SD,SDC;


if((RuleBase->rows_2==0)||(RuleBase->columns_2==0)||(RuleBase-
>matp[0][0].rows==0)||(RuleBase->matp[0][0].columns==0))
  {
  printf("\n!!!Error in infrb4d - Rulebase not defined properly");
  return(1);
  }

for(i=1;i<=NumCon;i++)
  {
  SD=0;
  SDC=0;

  for(j=0;j<DofsNZ[3].columns;j++)
```

```
{
for(k=0;k<DofsNZ[2].columns;k++)
    {
    for(l=0;l<DofsNZ[1].columns;l++)
      {
      for(m=0;m<DofsNZ[0].columns;m++)
        {
        register unsigned int r1=0,r2=0,c1=0,c2=0,crb=0;
        register float DOF=0,CON=0;
        MATRIX DofVec;

        r2=(unsigned int)DofsNZ[3].mat[0][j];
        c2=(unsigned int)DofsNZ[2].mat[0][k];
        r1=(unsigned int)DofsNZ[1].mat[0][l];
        c1=(unsigned int)DofsNZ[0].mat[0][m];

        if(r2>FV4NS-1)
            r2=FV4NS-1;
        if(r2<0)
            r2=0;
        if(r1>FV2NS-1)
            r1=FV2NS-1;
        if(r1<0)
            r1=0;
        if(c2>FV3NS-1)
            c2=FV3NS-1;
        if(c2<0)
            c2=0;
        if(c1>FV1NS-1)
            c1=FV1NS-1;
        if(c1<0)
            c1=0;

        crb=c1*NumCon+i-1;

        DofVec=matinit1(1,4,&DofVec);

        DofVec.mat[0][3]=Dofs[3].mat[0][r2];
        DofVec.mat[0][2]=Dofs[2].mat[0][c2];
        DofVec.mat[0][1]=Dofs[1].mat[0][r1];
        DofVec.mat[0][0]=Dofs[0].mat[0][c1];

        if(IF==1)
            DOF=MINIMUM(&DofVec);
        else if(IF==2)
            DOF=MAXIMUM(&DofVec);
        else if(IF==3)
            DOF=MEAN(&DofVec);
        else if(IF==4)
            DOF=ALGSUM(&DofVec);
        else if(IF==5)
            DOF=FUZOR(&DofVec,gamma);
        else if(IF==6)
            DOF=FUZUND(&DofVec,gamma);
        else if(IF==7)
            DOF=INFPROD(&DofVec);
        else
            printf("\nError in infrb4d - IF %d  not correct",IF);


        /*limit DOF*/
        if(DOF<0.0)
```

```
                DOF=0.0;
            if(DOF>1.0)
                DOF=1.0;

            CON=lookup4d(r2,c2,r1,crb,RuleBase);

            SD=SD+DOF;
            SDC=SDC+DOF*CON;

            matfree1(&DofVec);
            }
          }
        }
    }
  if(SD==0)
    {
    printf("\nSum of DOFS is zero - setting crisp to 0");
    crisp->mat[0][i-1]=0.0;
    return(1);
    }
  else
    crisp->mat[0][i-1]=SDC/SD;
  }

return(0);
}
```

Appendix C

Table of calibration data for the hot and cold inlet flowmeters.

$\overline{F_{cold}(t)}$ ml/s	V_{Flow} m^3	V_{Head} m^3	$F_{cold}(t)$ ml/s	$\overline{F_{hot}(t)}$ ml/s	V_{Flow} m^3	V_{Head} m^3	$F_{hot}(t)$ ml/s
10.35	0.0203	0.0905	46.15	19.92	0.0675	0.0958	28.26
19.69	0.0424	0.1359	63.14	39.15	0.0838	0.1028	48.04
43.07	0.0650	0.1135	75.17	84.22	0.1007	0.1185	99.12
52.72	0.093	0.1416	79.99	100.67	0.1343	0.1521	114.02
83.30	0.1141	0.1470	107.34	117.18	0.1443	0.1633	132.54
98.47	0.1327	0.1561	115.82	138.26	0.1499	0.1634	150.71
118.55	0.1487	0.1637	130.54				
134.12	0.1534	0.1557	146.89				
155.91	0.1534	0.1632	165.85				
175.75	0.1290	0.1466	199.77				
190.43	0.1482	0.1633	209.87				
203.85	0.1403	0.1520	220.30				
222.44	0.1428	0.1481	231.34				
231.80	0.1521	0.1613	245.88				

where $\overline{F_{cold}(t)}$ and $\overline{F_{hot}(t)}$ are the uncalibrated data for the cold and hot inlet flowmeters respectively,
$F_{cold}(t)$ and $F_{hot}(t)$ are the uncalibrated data for the cold and hot inlet flowmeters respectively and
V_{Flow} and V_{Head} are the volumes calculated by the flow and level methods as detailed in Section 4.2.

Appendix D

The following is the MATLAB macro code for the SIMULINK s-function for the physical model of the warm water process.

```
function [sys,x0]=tanksim4(t,x,u,flag,HeadInit,ToutInit,Atank,TauTC,Tamb)

% File :      TANKSIM3.M
% Created :    15.3.94
% By :        Stephen McCormac
%
% States ---> Tout, Head, Ttc
%           x(1), x(2), x(3)
%
% Inputs ---> Thot, Tcold, Qhot, Qcold, Voutpos
%           u(1), u(2),  u(3), u(4),  u(5),
%
% Outputs --> Ttc,   Qout,   Head,   CvA2
%           sys(1), sys(2), sys(3), sys(4)
%

% calculations for 1 and 3.
if ((flag==1)|(flag==3))
  Thot=u(1)-0.5;
  if ((x(1)-Tamb)>10)
    Kloss=3.5006e-4;
  elseif ((x(1)-Tamb)<3)
    Kloss=0;
  else
    Kloss=3.5006e-5;
  end;

  sqrt2g=4.4294;
  CvA2eqn=[-4.1739e-009  7.7567e-007 -2.0275e-007];
  CvA2=polyval(CvA2eqn,u(5));
  Qout=CvA2*sqrt2g*sqrt(x(2));
  %Qloss=Kloss/(Atank*x(3));
  if u(5)<=0
    Qout=0;
  end;
  ResHead=9.5e-2;    %head in tank below outlet valve-thus does
                     %not effect flow.
  ConeVol=3.019e-3;   %volume of cone at bottom of tank - m3
end

%initialisation of system
if flag==0

      sys=[3,0,4,5,0,0];
                              % [no. of cont. states, no. of discr. states
                              %  outputs, inputs, no of discontinuous
                              %  roots, no. of feedthrough ips to ops].

      x0=[ToutInit,HeadInit,ToutInit];  % initial conditions.

%evaluation of state derivatives
elseif abs(flag)==1
```

```
  %evaluation of first state - Tout
  if x(2)<0.05
    Level=0.05;
  else
    Level=x(2);
  end

  sys(1)=((Thot-x(1))*u(3)+(u(2)-x(1))*u(4)-
Kloss)/(ConeVol+(Atank*(Level+ResHead)));


  %evaluation of second state - Head
  Temp2=(u(3) + u(4) - Qout)/Atank;
  if x(2)>=1.75
    x(2)=1.75;
    if Temp2<0
      sys(2)=Temp2;
    else
      sys(2)=0;
    end
  else
    sys(2)=Temp2;
  end

  %evaluation of third state - Ttc
  Ttc=(x(1)-x(3))/TauTC;
  sys(3)=Ttc;

%evaluation of outputs
elseif abs(flag)==3

  %evaluation of output sys(1) - Tout
  sys(1)=x(3);

  %evaluation of output sys(2) - Qout
  sys(2)=Qout;

  %evaluation of output sys(3) - Head
  sys(3)=x(2);


  %evaluation of output sys(4) - CvA2
  sys(4)=CvA2;

end;
```

The following listing is the MATLAB macro code for the simulink icon of the artificial neural network model of warm water process as described in Sections 4.6 and 4.7.

```
function [sys,x0]=iccann1(t,x,u,flag,ts,pscale,tscale)

% File  :  iccann1.m
% Created :22.7.94
% By :     Stephen McCormac
%
% States --->
%
%
% Inputs --->
%
```

```
%
% Outputs -->
%
% Param. ---> ts, controller
%

% calculations for 1 and 3.


%initialisation of system
if flag==0

        sys=[0,4,2,9,0,0];
                                % [no. of cont. states, no. of discr. states
                                %  outputs, inputs, no of discontinuous
                                %  roots, no. of feedthrough ips to ops].

        x0=[0 0 0 0];     % initial conditions of state

%evaluation of continuous state derivatives
elseif abs(flag)==1

%evaluation of discrete state derivatives
elseif abs(flag)==2


  sys((2+1):2*2) = x(1:2);

  if abs(round(t/ts)-t/ts)<1e-6
    global w1 w2 w3 b1 b2 b3;
    u=u./pscale;
    sys(1:2)=tscale.*purelin(w3*logsig(w2*logsig(w1*u,b1),b2),b3);


  else
    sys(1:2)=x(1:2);
  end;


%evaluation of outputs
elseif abs(flag)==3
  sys=x((2+1):2*2);


%evaluation of next update time
elseif abs(flag)==4
  ns=t/ts;
  sys =(1 + floor(ns + 1e-13*(1+ns)))*ts;
end;
```

Appendix E - *Software Engineering Issues*

E.1. Introduction

E.1.1. General Introduction

This appendix discusses the issues concerning the development of the necessary software for simulation and control of the warm water process used in this research. This software can be categorised into five groups as follows :

1. *Fuzzy Logic Toolbox for MATLAB* - for the simulation of standard fuzzy controllers, self-organising controllers and supervised and unsupervised adaptive fuzzy models.

2. *Warm Water Process Utility Programs* - this software automates some of the necessary adjustments of the warm water process for ease of use. One such example is filling the tank to a certain level.

3. *ANSI C Matrix Software* - an ANSI C software representation of matrices with associated functions such as add, scalar multiplication and matrix multiplication. This provides a suitable data structure for the fuzzy control software needed for real time control of the warm water process.

4. *Artificial Neural Network Software* - ANSI C software implementation for the simulation of multi layer perceptron ANNs.

5. *ANSI C Fuzzy Functions* - this set of functions can be used for fast simulation of fuzzy logic algorithms within a compiled program. The other main application is that of real time control of the warm water process.

E.1.2. Overview of Chapter Structure

Section E.2 of this appendix describes the structure and use of the fuzzy toolbox for MATLAB. The set of utility programs for the warm water process is described in *Section E.3*. The ANSI C source code matrix representation and related functions are described in *Section E.4*. The next *Section E.5* iterates the implementation of the multi-layer perceptron neural network using ANSI C source code. An overview of the fuzzy functions in ANSI C format is contained in *Section E.6*. Finally speed and memory considerations for the fuzzy control software used in this research are summarised in *Section E.7*. The reader is referred to Chapter 2 for explanations of the fuzzy logic related terminology used throughout this appendix.

E.2. MATLAB Fuzzy Logic Toolbox

E.2.1. Introduction

MATLAB 4.0 contains no dedicated software for the simulation of fuzzy logic algorithms. Thus if the advantages of the MATLAB/SIMULINK simulation environment were to be utilised for the necessary simulation of adaptive fuzzy control within this research, it was clear that a *Fuzzy Control Toolbox* was required. After consideration of the requirements and the complexity of the necessary software, the following features were decided upon :

- *GUI* - Simple ***G***raphical ***U***ser ***I***nterface to increase the user friendliness of the software.
- *Fuzzy Membership Set Functions* - triangular, trapezoidal and Gaussian functions.
- *Inference Functions* - minimum, maximum, product, algebraic sum, fuzzy-and, fuzzy-or and mean functions.
- *Aggregation Functions* - minimum, maximum and mean functions.
- *Rulebase* - in a look-up table format.
- *Defuzzification Methods* - centre of gravity algorithms for both fuzzy membership function consequents and fuzzy singleton consequents.
- *SIMULINK Graphical Icons* - direct fuzzy controller, self-organising controller with a selection of performance indexes, supervised and non-supervised adaptive fuzzy models.
- *Fuzzy Controller Characteristic* calculation and a graphical representation thereof.
- *Link to ANSI C* - a link to the ANSI C fuzzy logic functions described in Section E.6.

The following Sections E.2.2 to E.2.8 describe the constituent parts of the fuzzy toolbox which was designed to meet all of the above listed criteria.

E.2.2. Graphical User Interface

During the design of the fuzzy toolbox most effort was invested in the functionality and not in user comfort. Although MATLAB 4.0 offers a broad spectrum of object orientated graphical functions, only a few are exploited by the fuzzy toolbox to improve the user friendliness. The most commonly used graphical function is *menu()* [112]. One example of the utilisation of this command is the main menu of the toolbox, as shown in Figure E.1. The user uses the mouse to select an option. The arguments of *menu(..)* then direct further program flow. The graphics functions contained in MATLAB 4.0 could be easily used to improve the visual quality and the user friendliness of the fuzzy tool box if further resources existed.

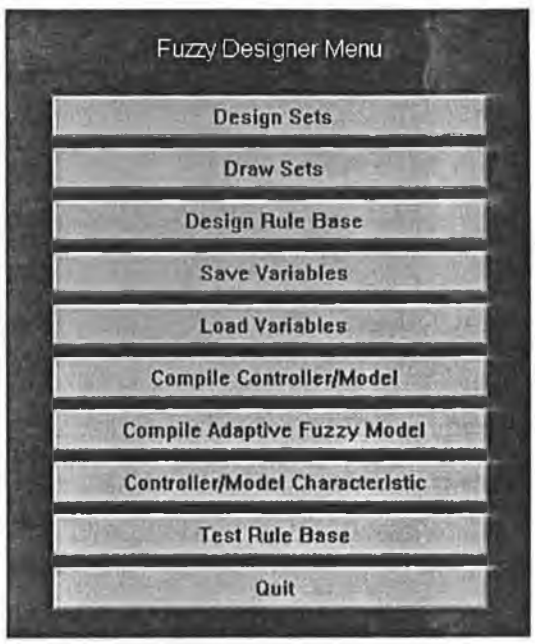


Figure E.1 - Main menu for the MATLAB fuzzy toolbox

E.2.3. Fuzzy Membership Set Functions

As described in Chapter 2 all crisp input variables of a standard fuzzy controller are converted to fuzzy variables through the process of fuzzification. For the fuzzy tool box, a simple and extendible storage medium for the fuzzy sets of a particular fuzzy variable was required. This medium was to contain the following information :

- *number of sets* for the variable,
- *minimum and maximum values* of the universe of discourse of the fuzzy variable and
- *individual fuzzy membership parameters* such as fuzzy membership set types and parameters.

The three most common functions for fuzzy membership sets were created for the fuzzy toolbox : *the triangular, the trapezoidal and the Gaussian functions.* In addition, a shouldering option for a set of fuzzy membership sets was offered. Figure

E.2 contains a plot of five shouldered triangular fuzzy membership sets over the universe of discourse [-1 , 1] taken directly from the fuzzy logic toolbox.

A two matrix data structure was chosen to represent the fuzzy membership set data for a given fuzzy variable. The first matrix, ***MATRIX 1***, contains general set parameters. The second matrix, ***MATRIX 2***, contains the parameters for the first order polynomials that describe the lines in the triangular and trapezoidal fuzzy sets. This pre-calculation increases the on-line processing speed of the fuzzification function.

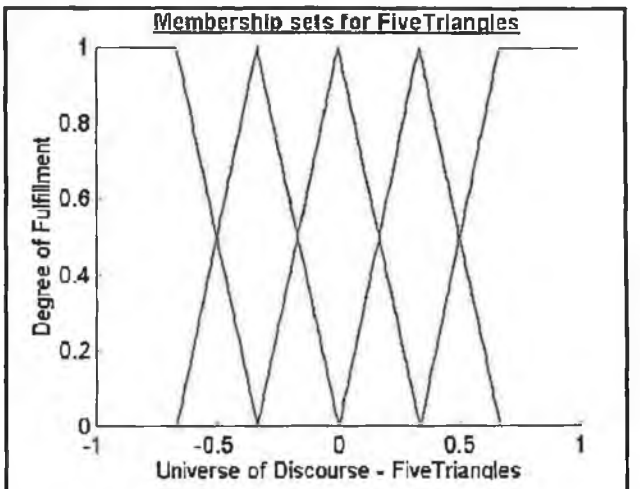


Figure E.2 - Plot of Fuzzy Membership Sets.

These two matrices are shown below, together with the definitions of their parameters:

MATRIX 1

n	min	max	–	–	–	–	–	–	–
type 1	.	.	.	.	.	.	.	.	label 1
.	.	.	.	.	.	.	.	.	.
triang	x1	x2	x3	-	y1	y2	y3	–	label
trap	x1	x2	x3	x4	y1	y2	y3	y4	label
gauss	c	v	–	–	h	–	–	–	label
.	.	.	.	.	.	.	.	.	.
type n	.	.	.	.	.	.	.	.	label n

MATRIX 2

$$\begin{bmatrix} set_{11}^{1} & set_{12}^{1} & set_{21}^{1} & set_{22}^{1} & set_{31}^{1} & set_{32}^{1} \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ set_{11}^{n} & set_{12}^{n} & set_{21}^{n} & set_{22}^{n} & set_{31}^{n} & set_{32}^{n} \end{bmatrix}$$

where

n	is the number of sets,
type	is an integer to indicate fuzzy membership set type, **1**=triangular, **2**=trapezoidal, **3**=Gaussian distribution,
min	is the minimum value of the universe of discourse,
max	is the maximum value of the universe of discourse,
label n	is the numerical index for the nth set,
x1 - x4	are the x axis parameters of the triangular and trapezoidal fuzzy membership sets,
y1 - y4	are the y axis parameters of the triangular and trapezoidal fuzzy membership set functions,
c,v,h	are the ***c***entre, ***v***ariance and ***h***eight parameters of the Gaussian fuzzy membership set functions and
$set^n_{11} \ldots . set^n_{32}$	are first order polynomial parameters to specify the lines in the triangular and trapezoidal fuzzy membership set functions.

The options *Design Sets* and *Draw Sets* in the main menu of the fuzzy tool box allow the user to firstly design the fuzzy membership sets and to plot them. Figure E.3 shows the menu obtained when the *Design Sets* option is chosen. The option *Default Values* creates a regularly spaced distribution of fuzzy membership sets along the specified universe of discourse.

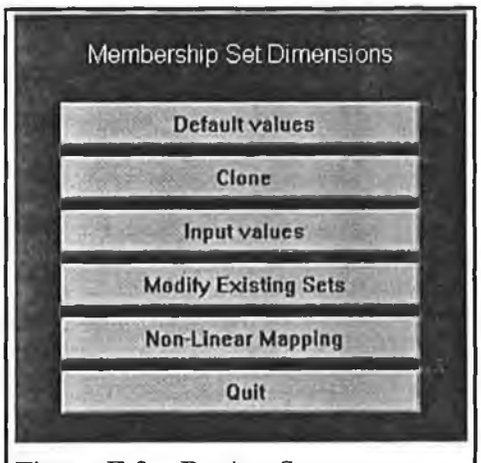


Figure E.3 - *Design Sets* menu.

Clone allows the copying of another fuzzy variable which saves time during definition of equivalent fuzzy variables. The choice of *Input Values* allows the exact specification of all set parameters. Should an existing fuzzy variable require alteration, *Modify Existing Sets* is to be activated. The option *Non-Linear Mapping* evaluates new triangular and trapezoidal fuzzy membership set parameters using the mapping equation (E.1) where *y* is user defined and *parameter* is a fuzzy membership parameter.

$$\text{Parameter}_{new} = (\text{Parameter}_{old})^{y} \qquad \text{(E.1)}$$

The user returns to main menu by choosing the *Quit* option.

E.2.4. Logical Operators

Logical operators are mathematical representations of linguistic operators such as *AND*, *OR* and *ELSE*. They are used as inference and aggregation functions within fuzzy algorithms. With reference to Chapter 2, where some of the many possible operators have been iterated, the fuzzy toolbox contains the *minimum*, *maximum*, *product*, *fuzzy-or*, *fuzzy-and*, *algebraic sum* and *mean* operators. All of these functions have been realised as user defined MATLAB functions and can thus be called from MATLAB and SIMULINK. All six of these functions can be used as Fuzzy Inference functions. The functions *minimum*, *maximum* and *mean* are available for the aggregation.

E.2.5. Rulebase Definition

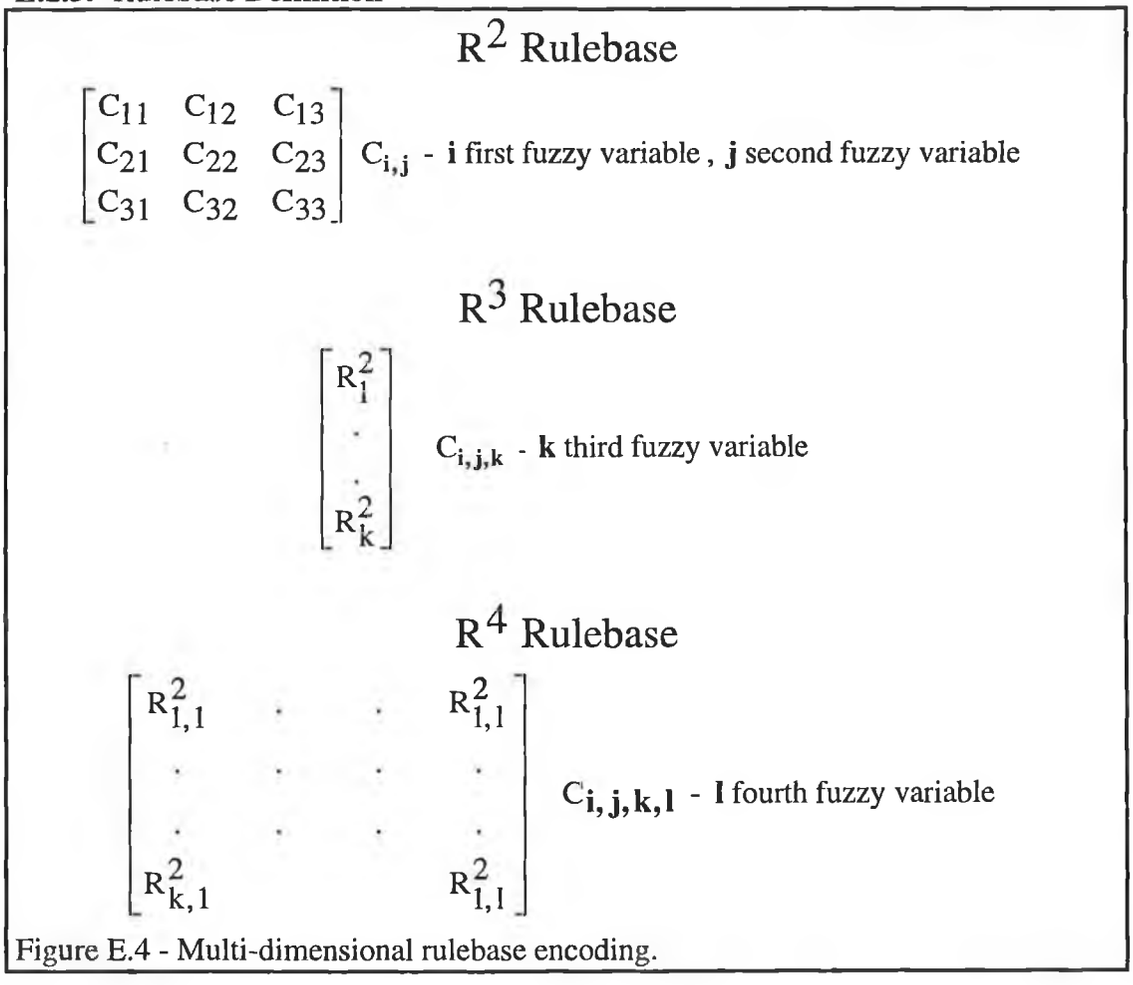


Figure E.4 - Multi-dimensional rulebase encoding.

Once the fuzzy variables with their corresponding fuzzy membership sets have been defined, the rulebase can be created. A lookup table format for the rulebase is chosen as this allows easy processing by MATLAB and simple graphical representation. Each element of the rulebase can contain one or more consequent values of a rule. This value is either a *fuzzy membership set index* or a *fuzzy singleton value*. The

index of a consequent is controlled by which antecedent fuzzy membership sets are active. As MATLAB cannot directly process matrices with more than two dimensions (R^2), fuzzy rulebases with more than two dimensions, i.e. with more than two input variables, require an encoding mechanism to translate to and from R^2 matrices. This encoding is graphically depicted in Figure E.4. For the example of a four dimensional rulebase $R^4\{X_1,X_2,X_3,X_4\}$ the complete rulebase has $X_2.X_4$ rows and $X_1.X_3$ columns where X_n is the number of membership sets and n the fuzzy variable index.

The main menu option *Design Rulebase* creates a matrix of zeros of the correct dimension when the user inputs the antecedent and consequent fuzzy variable names. The consequent values contained in each rulebase cell are then entered by the user either through direct matrix manipulation in MATLAB or with a text editor.

E.2.6. Defuzzification Methods

As described in Chapter 2 there are over thirty different defuzzification methods available to the user. The MATLAB fuzzy logic toolbox utilises only two of these - *centre of gravity for fuzzy membership and for fuzzy singleton consequents*. Both of these methods have been realised as MATLAB functions.

E.2.7. Fuzzy Controller and Fuzzy Models SIMULINK Icons

In order to allow the creation of user defined SIMULINK icons (*s-functions*) for the fuzzy controllers and fuzzy models to be simulated, the concept of a SIMULINK sub-routine was utilised. Two MATLAB commands are exploited for this purpose - *str2mat()* for the creation of the sub-routine and *eval()* for the evaluation of the sub-routine within the SIMULINK icon. The function *str2mat()* allows the user to create a matrix of strings. The function *eval()* evaluates a string given as an argument as a MATLAB command. Thus, by simply creating a string matrix and executing each row in sequence, a MATLAB subroutine can be executed. Such sub-routines can then be given as a SIMULINK icon argument and, using *eval()* within the *s-function* of the SIMULINK icon, executed. The main advantage of this method is the ease with which SIMULINK icons can be created for diverse purposes such as fuzzy control or fuzzy modelling.

Through application of the MATLAB sub-routine, the task of defining fuzzy controllers and fuzzy models was simplified to the creation of string matrices. These

string matrices are then entered as SIMULINK icon arguments and these icons can then be further manipulated within the SIMULINK environment.

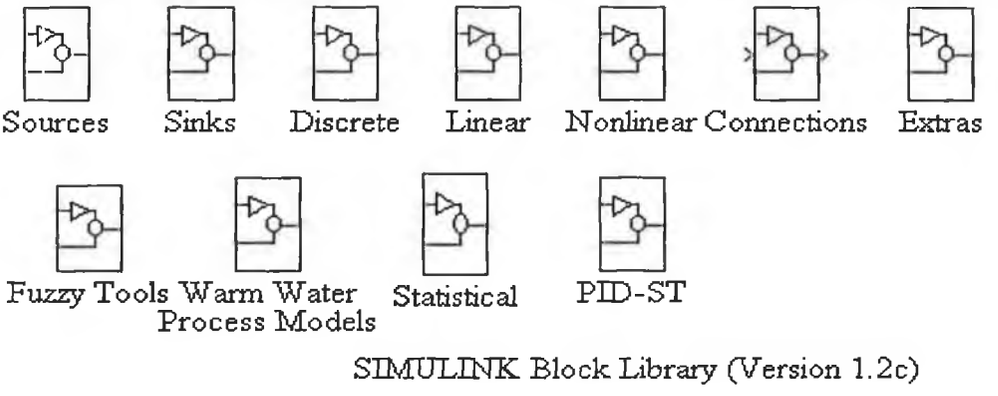


Figure E.5 - Main SIMULINK library showing the *Fuzzy Tools* icon.

There are two options in the main menu of the fuzzy logic toolbox for creating the sub-routine strings for the SIMULINK icons: *Compile Fuzzy Controller/Model* which allows the user to compile either a standard fuzzy controller or a non-adaptive fuzzy model and *Compile Adaptive Fuzzy Model*, which allows the user to create sub-routines for adaptive fuzzy models of the format described in Section 4.9. These adaptive fuzzy models which utilise the learning algorithm given by equation (4.25) and which is described in Section 4.9.2.5. All fuzzy and rulebase variables for these sub-routines must be previously defined in order for the SIMULINK icon to function correctly.

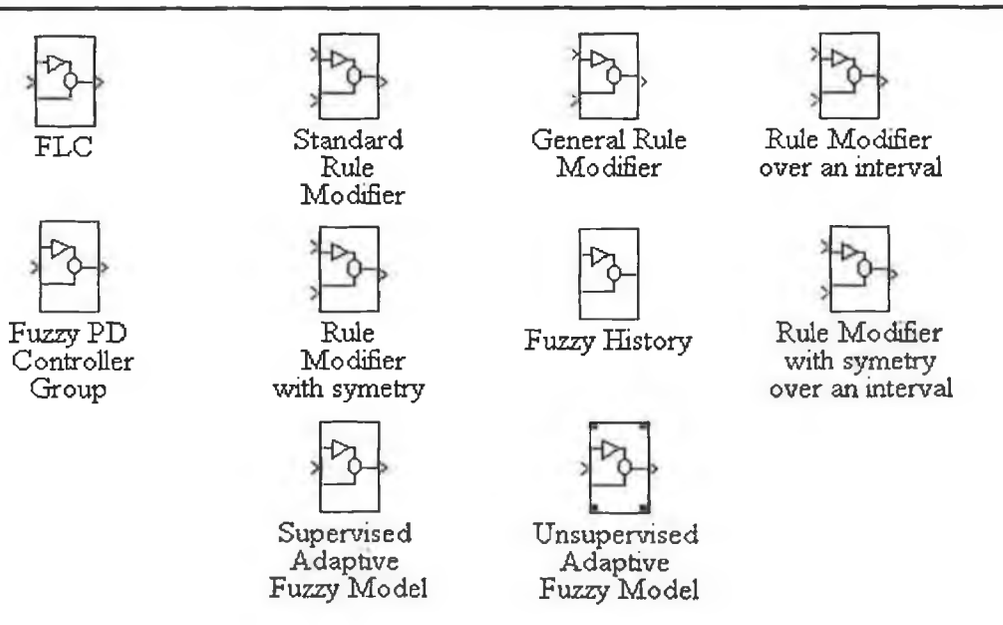


Figure E.6 - Fuzzy logic simulation tools for the SIMULINK environment.

A fuzzy toolbox icon *Fuzzy Tools* was thus created for inclusion within the SIMULINK main menu as shown in Figure E.5 . Figure E.6 shows the contents of this *Fuzzy Tools* icon when it is opened. The fuzzy toolbox icon contains icons for a standard fuzzy controller, self-organising controller with various performance indexes and rule modification algorithms and supervised and unsupervised fuzzy models. One example of these simulation icons is the *basic building block* for a standard fuzzy controller in SIMULINK - the icon entitled *FLC*. The four arguments for this block are :

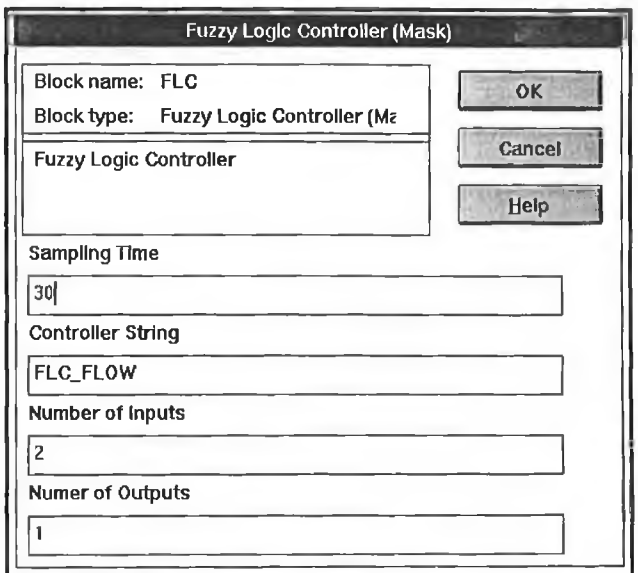


Figure E.7 - FLC Simulation Icon menu.

- the *sampling time* in seconds,
- the *fuzzy controller sub-routine string* which has been compiled by the fuzzy logic toolbox,
- the *number of inputs* of the fuzzy controller and
- the *number of outputs* of the fuzzy controller.

These four arguments are entered by the user by double clicking with the mouse on the icon and entering the values in the menu. This user defined menu is shown in Figure E.7.

E.2.8. Data Archiving

The two commands *Save Variables* and *Load Variables* listed in the fuzzy toolbox main menu allow the user to save and load fuzzy variables, rule base matrices, fuzzy controller and fuzzy model sub-routine strings to and from the hard disk. These variables can be defined as default variables by placing them in the file vardef1.m. After starting the fuzzy tool box, the user is prompted to decide whether or not the default variables contained in the file vardef1.m should be loaded into the MATLAB workspace.

E.2.9. Fuzzy Algorithm Testing Functions

There are two options in the fuzzy logic toolbox for testing a compiled fuzzy algorithm.

The *first option* calculates a characteristic surface for any two input variables of a fuzzy algorithm while holding any other arguments at constant values defined by the user. The user specifies the lower and upper limits and the number of steps for each variable. This function can be used to create a lookup table characteristic of a fuzzy controller. Such a lookup table characteristic could be constructed for a micro-controller implementation of a two input fuzzy controller with 8-bit ADCs by calculating a lookup table with 32 steps per input variable, giving a lookup table with the 1024 elements. Linear interpolation can be used between the 32 data points of the lookup table characteristic. Such an approach is suitable for fuzzy logic applications for use on hardware platforms with limited processing and memory resources.

The *second option* enables the user to enter values for the crisp input variables of the fuzzy algorithm, and the corresponding output value of the fuzzy algorithm is then calculated and outputted to the screen.

E.3. Warm Water Process Utility Programs

A set of six ANSI C utility programs were written in order assist the user in the control of the warm water process. The operation of all of these programs is terminated by hitting any key on the keyboard or when the level in the process reaction tank exceeds 170cm. The six functions together with a brief explanation of each are listed below:

- *Amplifier Calibration* - this program is used to calibrate the signal processing board described in Chapter 3. The six ADC integer values from each channel are displayed on the screen.

- *Physical Variable Observation* - this program displays the current values of the six ADC integer values and the calculated values of the physical variables on the screen. This can be used for checking the current values of the physical variables e.g. the temperature of the hot reservoir tank.

- *Heat Hot Reservoir Tank* - this program fully opens the hot inlet and outlet valves of the process reaction tank until the temperature of the hot reservoir tank reaches a user defined value.

- *Heat/Cool Process Reaction Tank* - the user enters a desired temperature and the hot or cold inlet flows are fully opened until the temperature of the process reaction tank has reached the desired value.

- *Fill Tank With Hot/Cold Water* - This program allows the user to fill the process reaction tank with specified values of the hot and cold inlet flows, with a defined outlet valve (0% - 100%) position assuming a linear outlet valve characteristic. The inlet flow and outlet flow valves are closed when the level of the tank reaches 170cm.

- *Empty Tank* - this program simply closes the inlet valves and fully opens the outlet valve until the outlet flow is zero for more than 30 seconds.

All of these utility programmes log the values of the six warm water process sensor values to a file entitled c:\utildata.asc.

E.4. ANSI C Matrix Representation and Related Functions

In order to ease the software engineering burden it is imperative that suitable data structures be utilised within a software program. As the fuzzy logic toolbox utilised the matrix structure found within MATLAB for the representation and simulation of fuzzy logic algorithms, the ANSI C source code for fuzzy logic functions was based on a data representation of a matrix. The ANSI C command *struct* was used to combine a two dimensional array of single precision floating point numbers together with unsigned integers to describe the size of the matrix. This data structure was then defined as **MATRIX** using the *typedef* ANSI C command. The ANSI C source code for the data type **MATRIX** is shown below :

```
type def struct {
  float huge **mat;
  unsigned int rows;
  unsigned int columns;
  } MATRIX;
```

The keyword *huge* allows this data structure to be used with any type of DOS memory model [92].

Dynamic memory allocation is used to both allocate memory to a matrix and free memory when a matrix is no longer required during program execution. Thus, in

order to define a matrix the function *matinit(number_of _rows, number_of_columns, &MATRIX_NAME)* is called and the ANSI C function *calloc* then allocates the necessary memory for the size of the matrix given and initialises all values to null. The memory taken up by a matrix is freed by calling the function *matfree(&MATRIX_NAME)* which deletes the matrix and allocates the freed memory to the general memory heap. In this way, matrices can be created and destroyed as they are needed during program execution. This helps to keep the size of compiled programs small as the number of pre-compilation defined variables is kept to minimum.

Based on this data type, a set of ANSI C functions were developed that allowed the manipulation of matrices. These functions include matrix addition and subtraction matrix multiplication and scalar multiplication of a matrix. This set of functions could easily be extended to include other functions such as matrix division and matrix inversion. The ANSI C source code for some of these MATRIX related functions is contained Appendix B.

E.5. ANSI C Representation of MLPs

The multi-layer perceptron ANN model of the warm water process, as described in Chapter 4, consists of neurons with non-linear transfer functions with weighted sums of inputs. The weights for the inputs to each layer of neurons are stored in matrix form within the MATLAB environment. Thus, the ANSI C matrix structure described in Section E.4 was utilised to create an ANSI C software implementation of the ANN model of the warm water process. In order to achieve this, some additional matrix functions were required for the non-linear neuron transfer functions. These extra functions calculate a neuron output with logarithmic sigmoid and linear transfer functions. The ANSI C source code for the ANN model of the warm water process is contained in Appendix B.

E.6. ANSI C Fuzzy Logic Functions

A set of ANSI C fuzzy functions used for real time control and fast simulation of fuzzy logic systems was created. The most important characteristic of this software is that it utilises the same matrix structure i.e. ***MATRIX 1*** and ***MATRIX 2***, for storage of the fuzzy variable parameters as the fuzzy logic toolbox for MATLAB which was described in Section E.2. This reuse of this matrix orientated parameter structure thus allows an easy transfer of a successfully simulated fuzzy controller or fuzzy model from the MATLAB/SIMULINK environment into an ANSI C program. Moreover,

should a rulebase be modified within a compiled ANSI C program, it can then be re-analysed in the MATLAB/SIMULINK simulation environment.

ANSI C software was written for fuzzy controllers and fuzzy models using rulebases of up to four dimensions. In order to allow the use of rulebases with dimensions higher than two (R^2), i.e. a standard matrix, the data type **MATRIX** as described in Section E.4 was augmented. Two new data types were created - **RB3D** for three dimensional rulebases and **RB4D** for four dimensional rulebases. As in the case of the MATRIX data type, functions were written to dynamically allocate and free memory for these rulebase matrices, see Section E.4. The ANSI C code source code for these rulebase variables can be found in Appendix B.

The fuzzy logic functions implemented are listed below :

- *Linguistic Operators* - Minimum, Maximum, Product, Fuzzy-And, Fuzzy-Or, Mean and the Algebraic Sum functions.

- *Fuzzification* - based on the MATLAB fuzzy logic toolbox fuzzification function utilising the same two fuzzy variable parameter matrices. Triangular, trapezoidal and Gaussian sets are handled.

- *Alpha-Cut* - sets to zero any degrees of membership less than the given argument.

- *Fuzzy Inference Functions* - these functions take the rulebase and the set of fuzzified input variables and deliver a crisp output value using centre of gravity defuzzification for fuzzy singleton consequents. These were written for two, three and four dimensional rulebases.

- *Adaptive Fuzzy Inference Functions (for adaptive fuzzy models)* - these functions take in the rulebase, the set of fuzzified input variables, the scalar value of the variable to be modelled and the parameters of the learning algorithm (4.25). The consequent variables of the activated rules are adapted and the adapted rulebase and the crisp output value are returned. There are two, three and four dimensional rulebase implementations of this function.

- *Complete Fuzzy Controller and Fuzzy Models* - these functions utilise combinations of the previously described functions to create complete fuzzy controllers and supervised adaptive fuzzy models.

The ANSI C code source code for some of these fuzzy functions can be found in Appendix B.

E.7. Speed and Memory Considerations

The software implemented during this research was based on experience gained during the course of fourteen months. The first step achieved for simulation was the *fuzzy logic toolbox for MATLAB*. This toolbox proved to quite effective in simulating fuzzy logic algorithms. The main disadvantage was the length of time required by the simulations. For a two dimensional rulebase with seven fuzzy membership sets per antecedent variable and fuzzy membership set consequents, the average processing time for one iteration in the MATLAB environment on a *i486 33Mhz IBM compatible PC with 8 MBytes RAM* was 0.32 seconds. This long processing time is attributable to the complexity of the software. For the simulation of standard fuzzy controllers such long simulation times were not a hindrance. But for the training of fuzzy model rulebases, the processing times were prohibitive. This becomes clearer when the example given for the fuzzy model of the mass flow behaviour of the warm water process in Section 4.9.4 is considered. The rulebase was trained for the equivalent of 20 million seconds at a sampling rate of 30 seconds. Had this been performed within the *MATLAB/SIMULINK* environment, the training time would have been 200000 seconds - two and a half days. At this point in the research it was decided to design the *ANSI C fuzzy logic software* to allow faster simulation using a compiled ANSI C program run on an IBM PC.

Iteration times for the same fuzzy model of the mass flow behaviour of the warm water process, as described in the previous paragraph, were reduced to a value in the region of 1ms for a single iteration running on an i486-DX2 50Mhz IBM compatible PC. This was a considerable improvement, reducing the processing time for the fuzzy functions during the training of mass flow fuzzy model to 667 seconds - just over eleven minutes. The main hindrance encountered while using the PC was the memory limitation presented by the DOS ANSI C compiler Turbo C version 2.0. This memory limit was approximately 610kBytes of RAM. Thus it was decided to port the software to a UNIX based workstation where a linear memory of 48MBytes was available.

The main hurdles to be overcome during the porting from the DOS to the UNIX operating system were the *conversion* of the DOS ASCII format to the UNIX ASCII format and the *removal* of all DOS related header files and data types. Once this was achieved, the fuzzy logic software benefited from the greater processing and memory

resources of the workstation platform, with fast fuzzy model training times and practically unlimited memory.

E.8. Conclusion

This appendix has served to give a *brief overview of the software* designed and implemented for this thesis. The source code of some of the more important ANSI C functions for matrices, multi-layer perceptron ANNs and fuzzy logic algorithms is contained Appendix B.

The three sets of ANSI C source code types i.e. matrices, ANNs and fuzzy logic algorithms could be extended and used for further research projects. A simple example would be the utilisation of the matrix software for the implementation of a state space pole placement controller. The ANN software could be augmented by an ANSI C implementation of the back-propagation learning algorithm in order to train an ANN with a compiled program, thus reducing the training time offered by the MATLAB Neural Network Toolbox, as described in Chapter 4. The ANSI C fuzzy software could be thoroughly optimised for speed of processing and extended by creating a library of learning algorithms for adaptive fuzzy models and perhaps by the implementation of relational matrix type rulebases.

References

[1] Tilli, T., "Fuzzy Logik, Grundlagen, Anwendungen, Hard- und Software", Franzis-Verlag GmbH & Co. KG München, 1992.

[2] Zimmerman H.-J., "Fuzzy Set Theory and its Applications", Kluwer Academic Publishers, 1991.

[3] Pedrycz, W., "Fuzzy Control and Fuzzy Systems", Research Studies Press LTD - John Wiley & Sons INC., 1993.

[4] Voit, F. , Voß, H.-J., Schnieder, E., Priebe, O., "Fuzzy Control versus Konventionelle Regelung am Beispiel der Metro Mailand", Automatisierungstechnik 42 (1994) 9, R. Oldenbourg Verlag, 1994, pp. 400-410.

[5] Sugeno, M. (Ed), "Industrial Applications of Fuzzy Control", North-Holland, 1985.

[6] Esogbue, A. O., Murrell, J. A., "Advances in Fuzzy Adaptive Control", Computers Math. Applic., Vol. 27, No. 9/10, 1994, pp. 29-35.

[7] Preuß, H.-P., Tresp, V., "Neuro-Fuzzy", Automatisierungstechnische Praxis 36, 1995, pp. 10-24.

[8] Werbos, P. J., "Neurocontrol and Fuzzy Logic: Connections and Designs", International Journal of Approximate Reasoning, No. 6, 1992, pp. 185-219.

[9] Batur, C., Srinivasan, A., Chan, C.-C., "Inverse Fuzzy Model Controllers", Proceedings of the American Control Conference, 1993, pp. 772-776.

[10] Iwasaki, T., Morita, A., "Auto-tuning Controller with Fuzzy Identification", Proceedings of the International Conference on Fuzzy Logic and Neural Networks, Iizuka Japan, 1990, pp. 401-404.

[11] Wang, L.-X., "Adaptive Fuzzy Systems", Prentice Hall, 1994.

[12] Dettmer, R., "Fuzzy Control - Engineering for Empiricists", IEE Review, January 1994, pp. 17-20.

[13] Zadeh, L. A., "Fuzzy Sets", Information and Control 8, 1965, pp. 338-353.

[14] Burkhardt, D. G., Bonissone, P. P., "Automated Fuzzy Knowledge Base Generation and Tuning", IEEE International Converence on Fuzzy Systems, 1992, pp. 179-188.

[15] Fei, J., Isik, C., "Adaptive Fuzzy Control via Modification of Linguistic Variables", IEEE International Converence on Fuzzy Systems, 1992, pp. 399-406.

[16] Nomura, H., Hayashi, I., Wakami, N., "A Learning Method of Fuzzy Inference Rules by Descent Method", IEEE International Converence on Fuzzy Systems, 1992, pp. 203-210.

[17] Procyk, T. J., Mamdani, E. H., "A Linguistic Self-Organising Process Controller", Automatica Vol. 15, 1979, pp. 15-30.

[18] Linkens, D. A., Abbod, M. F., "Self-Organising Fuzzy Logic Control for Real Time Processes", International Conference on Control '91, Edinburgh, 1991, pp. 971-976, Vol. 2.

[19] Moore, C. G., Harris C. J., "Indirect Adaptive Fuzzy Control", International Journal of Control, Vol. 56, No. 2, 1992, pp. 441-468.

[20] Moore, C. G., Harris C. J., Brown, "Intelligent Control: Aspects of Fuzzy Logic and Neural Nets", World Scientific, 1993.

[21] Angstenberger, J., "atp Marktanalyse : Software-Werkzeuge zur Entwicklung von Fuzzy Reglern", atp Automatisierungstechnische Praxis 35 (1993) 2, 1993, pp. 112-124.

[22] "Cubicalc Users Handbook", Hyper Logic Corporation, 1992.

[23] "TilShell User's Guide", TogaiInfralogic Inc., 1992.

[24] "FuzzyTech Benutzer Handbuch", Inform GmbH, 1992.

[25] "Real-Time Fuzzy Logic Block - User's Guide", Integrated Sytems Inc., 1991.

[26] Rossman, J., "Der Realismus kehrt ein - Die zweite Generation von Fuzzy-Chips wird abgespeckt", Elektronik 17, 1992, pp. 40-46.

[27] Dewitz, H. von., Kasper, C., Lieven, K., "Entwicklungen und Applikationen von Hardware-Lösungen mit integrierter Fuzzy-Logik", Ministerium für Wirtschaft, Mittelstand und Technologie des Landes Nordrhein-Westfalen, 1993.

[28] "FC110 Development System Users Manual V2.0.3, FC110DX-10E" December 1991, Togai Infralogic Inc.

[29] Tong, R. M., Beck, M. B., Lattens, A., "Fuzzy Control of the Activated Sludge Wastewater Treatment Process", Automatica, Vol. 16, 1980, pp. 659-701.

[30] Ostergaad, J. J., "Fuzzy Logic Control of a Heat Exchange Process", Fuzzy Automata and Decision Processes", 1977, p. 285.

[31] Yanunoba, S., Hasegawa, T., "Automatic Train Operation by Predictive Fuzzy Control", Control Theory Adv. Technol., 2, 3, 1986, p. 419.

[32] Altrock, C. v., Arend, H., Krause, B., Steffens, C., Behrens-Römmler, E., "Adaptive Fuzzy Control Applied to Home Heating System", Fuzzy Sets and Systems 61, 1994, pp. 29-35.

[33] Kluthe, R., "Clever geregelt - Das Potiential intelligenter Systeme am Beispiel der Kfz-Elektronik", Elektronik 26, 1994, pp. 91-96.

[34] Preuß, H., "Fuzzy Control - heuristische Regelung mittels unscharfer Logik - Teil 1", atp-Automatisierungstechnische Praxis 34, 1992, 4, pp. 176 - 183.

[35] Preuß, H., "Fuzzy Control - heuristische Regelung mittels unscharfer Logik - Teil 2", atp-Automatisierungstechnische Praxis 34, 1992, 5, pp. 239-246.

[36] Daugherity, W. C., Rathakrishnan, B., Yen, J., "Performance Evaluation of a Self-Tuning Fuzzy Controller", IEEE International Conference on Fuzzy Systems, 1992, pp. 389-397.

[37] Bitmead, R. R., "Setting our house in order", IEEE Controls Systems Magazine, Vol.13, No.3, June 1993.

[38] Lee, C. C., "Fuzzy Logic in Control Systems, Part I and II", IEEE Transactions on Systems, Man and Cybernetics, Vol. 20, no. 2, 1990.

[39] Kiendl, H., "Invarianzforderungen für Inferenzfilter", 4. Workshop "Fuzzy Control" des GMA-UA 1.4.2, Forschungsbericht Nr. 0194, Dortmund, 1994.

[40] Protokol des GMA Unterausschußes 4.5.1, "Fuzzy Control", Frankfurt am Main, October, 1994.

[41] Liu, M., H., "Fuzzy-Modellbildung und Ihre Anwendungen", VDI Berichte 1113, GMA-Aussprachetag "Fuzzy Control", Langen, March 1994.

[42] Tong, R. M., "The Construction and Evaluation of Fuzzy Models", Advances in Fuzzy Set Theory and Applications, North-Holland Publishing Company, 1979.

[43] Takagi, T., Sugeno, M., "Fuzzy Identification of Systems and Its Applications to Modelling and Control", IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-15, No. 1, 1984.

[44] Küpper, K., "Self Learning Fuzzy Models Using Stochastic Approximation", 3rd IEEE CCA, Glasgow, Scotland, 1994, pp. 1723 -1728.

[45] Sugeno, M., Kang, G.T., "Fuzzy Modelling and Control of a Multilayer Incinerator", Fuzzy Sets and Systems 18, 1986, pp. 329-346.

[46] Bare, W. H., Mullholland, R. J., Sofer, S. S., "Design of a Self-Tuning Rule Based Controller for a Gasoline Refinery Catalytic Reformer", IEEE Transactions on Automatic Control, Vol. 35, No. 2, 1990.

[47] Isaka, S., Sebald, A. V., Karimi, A., "On the Design and Performance Evaluation of Adaptive Fuzzy Controllers", Proceedings of the 27th Conference on Decision and Control, Austin, Texas, USA, 1988, pp. 1068-1069.

[48] Zhao,-Y. Z., Tomizuka, M., Sagara, S., "A Fuzzy Tuner for Fuzzy Logic Controllers", Proceedings of the 1992 American Control Conference, pp 2268-72 vol. 3.

[49] Sugiyama, K., "A Rule-Based Self-Organising Controller", Fuzzy Computing, Elsevier Science Publishers, 1988, pp. 341-353.

[50] Shao, S., "Fuzzy Self-Organizing Controller and its Application for Dynamic Processes", Fuzzy Sets and Systems 26, 1988, pp. 151-164.

[51] Wakileh, B. A. M., Gill, K. F., "Robot Control Using Sef-Organising Fuzzy Logic", Computers in Industry, Netherlands, 1990, pp. 175-186.

[52] Ho, J. M., Lin, S., "A Learning Algorithm for Fuzzy Self-Organising Controller", IEEE International Workshop on Intelligent Motion Control, Bogazici University, Istanbul, 1990.

[53] Farbrother H. N., Stacey B. A., Sutton R., "Fuzzy Self-Organising Control of a Remotely Operated Submersible", International Conference on Control '91, Edinburgh, 1991, pp. 499-504.

[54] Spinrad, M. D., "Self-Organising Fuzzy Control", Advances in Instrumentation and Control, Vol. 46, Proceedings of the ISA/91 International Conference and Exhibition, 1991, pp. 1247-1260 vol. 2.

[55] Spinrad, M. D., "Self-Organising Fuzzy Control", Proceedings of the ISA/92 International Conference and Exhibition, 1992, pp. 1161-1171.

[56] Zhang, B. S., Edmunds, J. M., "Self-Organising Fuzzy Logic Controller", IEE Proceedings D (Control Theory and Application), Vol. 139, No. 5, 1992, pp. 460-464.

[57] Song, J. J., Park, S. "A Fuzzy Dynamic Learning Controller for Chemical Process Control", Fuzzy Sets and Systems 54, 1993, pp. 121-133.

[58] Nomura, H., Hayashi, I., Wakami, N., "A Self-Tuning Method of Fuzzy Control by Descent Method", Proceedings of the 4th IFSA Congress, Engineering, pp. 155-158.

[59] Batur, C, Kasparian, V., "Fuzzy Adaptive Control", International Journal of Systems Science, Vol. 24, No. 2, 1993, pp. 301-314.

[60] Chen, Y.-Y., Lin, K.-Z., Hsu, S.-T., "A Self-Learning Fuzzy Controller", IEEE International Conference on Fuzzy System, 1992, pp. 289-296.

[61] Wang, L.-X., "Stable Adaptive Fuzzy Control of Nonlinear Systems", IEEE Transactions on Fuzzy Systems, Vol. 1, No. 2, 1993, pp. 146-155.

[62] Mallampati, D., Shenoi, S., "Self-Organising Fuzzy Logic Control", Knowledge Based Systems and Neural Networks:techniques and Applications, 1990, pp. 271-282.

[63] Stipanicev, D., De Neyer, M., Gorez, R., "Self-tuning Self-organizing Fuzzy Robot Control", IFAC Robot Control (SYROCO '91), 1991, pp. 171-176.

[64] Maeda, M., Sato, T., Murakami, S., "Design of the Self-Tuning Fuzzy Controller", Proceedings of the International Conference on Fuzzy Logic and Neural Networks, Iizuka, Japan, 1990, pp. 393-396.

[65] Graham, B. P., Newell, R. B., "Fuzzy Adaptive Control of a First-Order Process", Fuzzy Sets and Systems 31, 1989, pp. 47-65.

[66] Ling, C., Edgar, T.F., "A New Fuzzy Gain Scheduling Algorithm for Process Control", Proceedings of the 1992 American Control Conference, Chicago, 1992, pp. 2284-2290 vol. 3.

[67] Klein, A., "Adaption eines Zustandsreglers mit Hilfe der Fuzzy-Set-Logik", O+P - Ölhydraulik und Pneumatik, 35, Nr. 8, 1991, pp. 605-612.

[68] Klein, A., Backé, W., "Optimisation of a State-Space-Controller by Fuzzy-Logic", 5th Bath Int. Fluid Power Workshop, 1992.

[69] Correspondence from Hanley Controls Ltd., dated 29.04.1991.

[70] "VarioPak-28000, A Microflow Valve with Actuator Having a Built-in Cv Adjuster", Instructions, Instruction No. EH 4500 E, Rev. B. 09/89, 1989.

[71] "Sensycon Control Valves - Group 23/16", DL 23-16b.

[72] Correspondence with Hanley Controls Ltd., dated 30.01.1992.

[73] "Series 35002 Camflex II Valve - Instructions", Masoneilan/Dresser, Instruction No. EF 5000E, Rev. B09/89, 1989.

[74] "Series 4600 Pneumatic Positioner - Instructions", Masoneilan/Dresser, Instruction No. EF 2000E.

[75] "Sensors and Systems for Process Management", Bush Beach Engineering Limited, 1991.

[76] "Flowmeters for Pipes up to 40mm", Perflow Instruments Ltd., Stock Sheet 2.

[77] "Instruction - 82 Series Turbine Flowmeter", Foxboro, MI 019-116, 1986.

[78] "Instruction - PA420 Analog Amplifier", Foxboro, MI 019-216, 1985.

[79] Sears, F. W., Zemansky M. W., Young H. D., "College Physics" Fifth Edition, Addison Wesley, 1980, pp. 246.

[80] Bush Beach Engineering Ltd, "Orifice Bore Calculation", Correspondence from Hanley Controls Ltd., dated 23.12.1991.

[81] "843 d/p Cell Transmitter", Foxboro, PSS 2A-1A15 A, 1987.

[82] "KSR Kuebler Level Sensors", Catalogue 1001, KSR Kuebler Control Engineering Limited.

[83] "Mounting and Operating Instructions Measuring Transformer for Mounting on Mounting Rails Type: MUA", KSR Kuebler Control Engineering Limited.

[84] "Instruction 8000 Series Pulsed dc Magnetic Flowmeter with Remotely-mounted Transmitter Styles A and B", Foxboro, MI 021-361, 1989.

[85] "Instruction 8000 Series Pulsed dc Magnetic Flowmeter - Configuration and Operation", Foxboro, MI 021-363, 1987.

[86] "PCL-812PG ADC Analog to Digital Conversion Card - Instructions", Advantech Ltd, 1993.

[87] "PCL-726 DAC Digital to Analog Conversion Card - Instructions", Advantech Ltd, 1993.

[88] "Operational Amplifier TI-071", Texas Instruments Inc., Linear IC Data Book, 1993.

[89] Jacob, M., Grabel, M., McGraw-Hill, "Microelectronics", International Editions, Electrical and Electronic Engineering Series, 1987.

[90] Correspondence with J. Whelan, B.Eng., Department of Electronic Engineering, Dublin City University.

[91] Eggebrecht, L. C., "Interfacing to the IBM Personal Computer", SAMS, Second Edition, 1990.

[92] "The Waite Group's Turbo C++ Bible", The Waite Group, 1993.

[93] "Borland Turbo C v2.0 Reference Guide", Borland Inc., 1987.

[94] "Borland Turbo C v2.0 User's Guide", Borland Inc., 1987

[95] Irie, B., Miyake, S., "Capabilities of Three-layered Perceptrons", Proceedings of the IEEE Int. Conf on Neural Networks, 1988, pp. 641-648.

[96] Kosko, B., "Neural Networks and Fuzzy Systems", Prentice Hall, 1992.

[97] Mott, R. L., "Applied Fluid Mechanics", MacMillan Publishing Company, 1994.

[98] "MATLAB Reference Guide", The MathWorks, Inc., 1992, pp. 380 - 382.

[99] Åstrom, K. J., Wittenmark, B., "Computer Controlled Systems", Prentice-Hall International Editions, 1990.

[100] Hannah, J., Hiller, M. J., "Applied Mechanics", Pitman Publishing Limited, 1971.

[101] Wasserman, P. D., "Advanced Methods in Neural Computing", Van Nostrand Reinhold, 1993.

[102] Berenji, H., "Neural Networks and Fuzzy Logic in Intelligent Control", Proceedings. 5th IEEE International Symposium on Intelligent Control 1990, pp. 916-20 vol. 2.

[103] Rummelhart, D.E., Hinton, G.E., Williams, R.J., "Learning Internal Representations by Error Propagation", Parallel Distributed Processing : Exploring the Microstructure of Cognition, Volume 1: Foundations, Bradford Books/MIT Press, 1986.

[104] Widrow, B., Lehr, M. A., "30 Years of Adaptive Neural Networks: Perceptron, Medaline, and Backpropagation", Proceedings of the IEEE, Vol. 78, No. 9, 1990, pp. 1415-1442.

[105] "MATLAB Neural Network Toolbox User's Guide", The MathWorks Inc, 1994.

[106] "SIMULINK: Dynamic System Simulation Software - User's Guide", The MathWorks, Inc., 1991-1993.

[107] "MATLAB Identification Toolbox User's Guide", The MathWorks Inc, 1992.

[108] Küpper, K., "Modellbildung mittels eines selbstlernenden Fuzzy-Systems", Forschungsbericht Nr. 8/94, Gerhard Mercator Universität Duisburg Gesamthochschule, 1994.

[109] Lai, J.-Y., Lin, Y.-C., "Fuzzy Model-Based Control of a Pneumatic Chamber", Proceedings of the American Control Conference, 1993, pp. 1162-1166.

[110] "MATLAB User's Guide", The MathWorks, Inc., 1993, pp. 2-156 - 2-183.

[111] Banyasz, C., Keviczky, L., "Direct Methods for Self-tuning PID Regulators", Proc. 6th IFAX Symp. on Identification and System Parameter Estimation, Washington DC, June 1982.

[112] "MATLAB Reference Guide", The MathWorks, Inc., 1992, pp. 328.