Wavelet-Based Identification and Control of Chemical Processes

THESIS

Submitted in partial fulfillment
of the requirements for the degree of
DOCTOR OF PHILOSOPHY

By

HARE KRISHNA MOHANTA

Under the Supervision of
Prof. R. K. GUPTA

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE
PILANI (RAJASTHAN) INDIA
2006

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI
RAJASTHAN, INDIA

CERTIFICATE

This is to certify that the thesis entitled **“Wavelet-Based Identification and Control of Chemical Processes”** and submitted by **Hare Krishna Mohanta** ID.No.**1999PHXF010** for award of Ph.D. Degree of the Institute, embodies original work done by him under my supervision.

Signature in full of the Supervisor	
Name in capital block letters	**Dr. RAM KRISHNA GUPTA**
Designation	Principal Icfai Institute of Science and Technology Bangalore, India

Date: 28.10.2005

ACKNOWLEDGEMENTS

I express my profound sense of gratitude and appreciation for my thesis supervisor *Prof. R.K. Gupta*, Principal, Icfai Institute of Science and Technology, Bangalore and former Professor and Group Leader, Chemical Engineering Group, BITS Pilani, for his valuable guidance, keen interest, constant encouragement and constructive advice throughout the period of this research work, without which, this work would not have seen the light of the day. His inspiration, humbleness, immense patience and the liberty he has given at work enabled me to present my thesis in this form. It has been a great pleasure and privilege for me to work under his guidance.

I thank the members of the Doctoral Advisory Committee, *Prof. R. P. Vaid*, Professor Chemical Engineering Group and *Dr. Amiya Kumar Jana*, Lecturer, Chemical Engineering Group, for their kind suggestions and moral supports.

My special thanks go to *Prof. S. Venkateswaran*, Vice-Chancellor, B.I.T.S. Pilani, for giving me an opportunity to do research at the Institute. I am thankful to *Prof. L.K. Maheshwari*, Pro-Vice-Chancellor and Director, B.I.T.S. Pilani, *Prof. K.E. Raman*, Deputy Director (Administration), *Prof. V.S. Rao*, Deputy Director (Off-Campus Programme), *Prof. A.K. Sarkar*, Dean Instruction Division and Faculty Division I, *Prof. B.V. Babu*, Assistant Dean Engineering Services Division and Group Leader, Chemical Engineering Group for providing the necessary infrastructure and other facilities.

My sincere thanks are due to *Prof. Ravi Prakash*, Dean, Research and Consultancy Division for his support, kindness, affection and encouragement during the research work. I thank *Dr. S.D. Pohekar* of Research and Consultancy Division for his help.

I extend special thanks to *Prof. T.N.S. Mathur* for willfully taking the trouble to correct the initial drafts of this thesis and for providing constant encouragement. My thanks are due to *Prof. B.R. Natarajan, Dr. S.D. Manjare, Mr. Basudeb Munshi, Mr. Rakesh Angira* and *Dr. (Mrs.) Madhusree Kundu*, for their encouragement and support. I am thankful to *Mr. Pratik N. Sheth, Mr. Amol A. Deshpande* and *Mr. Nikhil Prakash* for helping me in the research work. I thank my colleagues *Mr. Suresh Gupta, Ms. Smita Raghuvanshi, Dr. Kailash L. Wasewar, Dr. Arvind K. Sharma, Mr. Ashish M. Gujarathi, Dr. B.B. Gulyani, Dr. Rajan Agarwal, Mr. Sushil Kumar* and other members of Chemical Engineering

Group for providing a cohesive, conducive and friendly atmosphere in the Department. I also thank my friends and colleagues, *Mr. S.B. Mishra*, *Dr. A. K. Giri*, *Mr. Suvendu Sāhoo* and *Dr. Srikanta Routroy*, *Mr. B.K. Rout*, *Dr. M.S. Dasgupta* and *Dr. L. Beherā* for their support.

I express my sincere thanks to *Mr. Jangvir*, *Mr. Babulal*, *Mr. Shankar Mal*, *Mr. Jeevan Lal*, and *Mr. Ashok*, *Sri Mathuram ji*, *Sri Jamunadhar ji* for their cooperation during my Ph.D. work. I am thankful to *Mr. Tulsi Ram Sharma* and *Mr. S.S. Sharma* for their help during the experimentation. I would like to thank members of Reprography, Xerox and Printing Sections for their prompt services.

I thank *Dr. Harish Chandra*, *Prof. R.C. Jain*, *Ms. Shikha Tripathi* and *Mr. K.K. Gupta* and *Mr. Vinod R. Upadhyay* for helping me in understanding the concepts of wavelets.

I extend my sincere thanks to *Mr. K. Hari Babu* and *Mr. M.E.S. Vidya Sagar* for their help in correcting the software codes and in thesis preparation. I thank *Mr. Rajan Viswaranjan* for carefully taking the photographs of the experimental setups.

I express my special thanks to *Dr. A. K. Ghoshal*, Assistant Professor, Chemical Engineering Department, I.IT. Guwahati for his constant encouragement and support.

I am thankful to *H.G. Shyāmahari Dās*, *H.G. Harihara Dās*, *H.G. Vaikuntha Dās*, *Sri Nrsimha Dās*, *Bkt. Sudhir* and *Bkt Bablu* for providing a spiritual atmosphere in Pilani. My special thanks go to *Sri Ravinder Yadav* and his wife *Anitā Mātāji*, *Dr. Suryakanta Maharana* and *his mother* for their care, love, affection and help to my family during the time of difficulties. I am thankful to *Manoj Prabhu*, *Sitānāth Prabhu* and *Bhaskar Bhāi*.

I am deeply indebted to my parents, sisters and in-laws for their constant encouragement, patience, and moral support. I would like to record my special affection and thanks to my wife *Lakshmipriyā*, whose constant persuasion and moral support has been a source of inspiration to me. I also thank my son *Gopāl*, daughter *Vishākhā* and nephew *Kānā* for cheerfully sacrificing the time that rightfully belonged to them.

Last but not the least, I humbly acknowledge that I owe my ability in completing the research work to Lord Sri Krishna through my spiritual master *His Holiness Bhaktiswarup Dāmodar Swāmi*. I offer my humble obeisances at His Lotus feet.

HKMohanta

(Hare Krishna Mohanta)

Dedicated to

His Divine Grace

A.C. Bhaktivedānta Swāmi Prabhupāda

Founder-Āchārya of the International Society for

Krishna Consciousness (ISKCON)

CONTENTS

Chapters

ABSTRACT

Wavelet-based identification and control techniques are applied to two pilot plant processes, namely the heat exchanger process and the liquid level process. The heat exchanger process is identified first by classical least-square method and then by wavelet-based least-square method using closed-loop input-output raw noisy data. The parametric models are found to be second order difference equations. These models are tested for predicting 100 output values from a different set of input-output data. The Root Mean Square Error (RMSE) value with respect to the denoised output is 0.7991 in case of classical least-square identification (CLSI) and is 0.3442 in case of wavelet-based least-square identification (WLSI). The liquid level process is also similarly identified from the closed-loop input-output raw noisy data using CLSI and WLSI. In this case the parametric models are found to be first order difference equations. These models are tested for predicting 100 output values from a new set of input-output closed-loop experimental data. In case of CLSI the RMSE value is found to be 2.3038 whereas in case of WLSI the RMSE value is found to be 1.6120. Thus the parametric models obtained by WLSI give better prediction than those obtained by CLSI for both the example processes.

The control objective in case of the heat exchanger process is to keep the cold water outlet temperature at a desired value. The First-Order-Plus-Dead-Time (FOPDT) model of the heat exchanger process is obtained from the process reaction curves by giving a step change in the hot water flow rate (manipulated variable) keeping cold water flow rate (disturbance) constant and giving a step change in cold water flow rate keeping hot water flow rate constant. The Cohen-Coon and the Ziegler-Nichols settings for digital PID control (using computer) of the heat exchanger process are obtained using these models. The closed-loop responses of the process for a step change of 10°C in the cold water outlet temperature keeping cold water flow rate constant are used to measure the performances of the conventional PID controllers with different settings. The integral of square error (ISE) over a period of 300 seconds is utilized to judge the performances of the controllers. Using Cohen-Coon and Ziegler-Nichols settings in the digital PID controller the ISE values are found to be 2751 and 2218, respectively. The controller

performance is further improved by using the settings obtained from on-line trial and error (OLTE) method. In this case ISE value is found to be 1316.

The dynamic matrix controller (DMC) for the heat exchanger process has been designed from the step response model of the process. The step response coefficients are obtained from the open loop responses while giving a unit step change in the hot water flow rate, keeping cold water flow rate constant. These coefficients in deviation form are used to construct the dynamic matrix. The OLTE method is used to tune the DMC. The tuning parameters in DMC are NP (prediction horizon), NC (control horizon) and NT (model or truncation horizon) and MSP (move suppression parameter). The same criterion as that is used in PID control is used to judge the performance of the DMC. The performance of the DMC is found to be better than that of the conventional PID controller as an ISE value of 817 is obtained while implementing DMC for the same step change in the setpoint.

The wavelet-based dynamic matrix controller (WDMC) is developed within the framework of DMC. Two changes are made in the structure of DMC to obtain the WDMC: Firstly, the dynamic matrix is continuously updated using WLSI. Secondly, wavelet-based blocking and condensing (B & C) techniques are applied to reduce the size of the dynamic matrix without deteriorating the performance. By the first modification the step response model is continuously updated to reduce the mismatch between the true process and the mathematical model. By the second modification the controller robustness is improved and the computation time is also reduced. The tuned parameters in case of DMC are used in case of WDMC. The additional tuning parameters in case of the WDMC are the identification horizon (NI), blocking design parameter (d_u) and condensing design parameter (d_y). For the heat exchanger process the identification is conducted at 20th, 25th, 30th, …. 100th seconds using WLSI. The parameters d_u and d_y are tuned through OLTE method. The WDMC is implemented in the heat exchanger process and its performance is judged by the same criterion used in case of PID and DMC controllers. It is observed that the WDMC gives better performance than DMC as in this case the ISE value obtained is 730.

The performance of the WDMC is further compared with the Modified 7×7 Fuzzy Logic controller (FLC) developed by Dasgupta and Gupta (1998) for the same heat exchanger

process in identical operating conditions. It is observed based on the ISE values that the proposed WDMC performs better than the FLC developed by Dasgupta and Gupta (1998).

Regulatory controls of the heat exchanger process using conventional PID, DMC and WDMC are also studied for a step change of 100 cc/min in the cold water flow rate. The ISE values obtained in a period of 600 seconds are 2487, 1648 and 775 for PID, DMC and WDMC, respectively.

The control objective in case of the liquid level process is to keep the liquid level in the process tank at a desired value. The FOPDT model of the liquid level process is obtained from the process reaction curves by giving a step change in the inlet water flow rate (manipulated variable) keeping outlet water flow rate (disturbance) constant. Using similar approach as that for the heat exchanger process the conventional PID, DMC and WDMC are developed for the liquid level process. The closed-loop response of the process for a step change of 30mm in the water level in the tank keeping outlet water flow rate constant is used to measure the performances of different controllers. The ISE values in a period of 300 seconds are found to be 6580, 5725 and 5140 for PID, DMC and WDMC, respectively. In regulatory problem a pulse change of –20mm is given in the level. The ISE values are found to be 4811, 4259 and 3459 with PID, DMC and WDMC controllers, respectively. The ANN controller developed by Dasgupta (2004) gives ISE value of 4165 for the same regulatory control. It is observed from this study that the developed WDMC outperforms the PID, DMC, FLC and ANN controllers in terms of minimum ISE value.

KEY WORDS

Wavelet; WT; FOPDT model; PID; OLTE tuning; B & C; CLSI; WLSI; DMC; WDMC.

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

Abbreviations

AD/DA	Analog to digital and Digital to Analog converter
ANN	Artificial Neural Networks
B & C	Blocking and Condensing
C-C	Cohen-Coon settings
CLSI	Classical Least-Square Identification
CO	Controller output
CPU	Central Processing Unit
CSTR	Continuous Stirred Tank Reactor
CWT	Continuous Wavelet Transform
DAC	Digital to Analog Converter
DDC	Direct digital controller
DMC	Dynamic Matrix Control/Controller
DWT	Discrete Wavelet Transform
FLC	Fuzzy Logic Controller
FOPDT	First-order-plus-dead-time
FT	Fourier Transform
HWFR	Hot Water Flow Rate
IMC	Internal Model Controller
ISE	Integral of square error
MIMO	Multiple-Input Multiple-Output
MPC	Model Predictive Control/Controller
MRA	Multiresolution Analysis
MSP	Move Suppression Parameter
NMPC	Nonlinear Model Predictive Control/Controller
NC (M)	Control horizon
NI	Identification horizon
NP (P)	Prediction or projection horizon
NT	Truncation or model horizon

ODE/PDE	Ordinary / Partial Differential Equations
OLTE	Online-trial-and-error tuning
OHT	Overhead Tank
PID	Proportional-Integral-Derivative
PRC	Process Reaction Curve
QDMC	Quadratic Dynamic Matrix Controller
RMSE	Root mean square error
SD	Standard Deviation
SISO	Single-Input Single-Output
STFT	Short Time Fourier Transform
WLSI	Wavelet-based Least-Square Identification
WDMC	Wavelet-based Dynamic Matrix Control/Controller
WNN	Wavelet Neural Networks
WT	Wavelet Transform
Z-N	Ziegler-Nichols settings

Symbols

A	Dynamic matrix.
$\underline{\underline{A}}$	Data matrix, which contains measured values of the output and input at different instants.
$a(i)$	Unit step response functions of the system.
a_k	Parameters to be identified (coefficient to output, x)
b_k	Parameters to be identified (coefficient to input, m)
$c_{m,n}$	Projection on a scaling function at scale m and translated by n.
c_j	Projection of the function on a scaling function at scale j.
c_ψ	A constant that is a function of the wavelet.
$c(t)$	Controller output for the PID controller.
c_s	Controller bias value.
d	Disturbance in the process.
d	Vector of detailed components.

d_m	Wavelet coefficient at scale m.
d_j	Projection of the function on a wavelet function at scale j.
$\mathrm{d_u}$	Blocking design parameter.
$\mathrm{d_y}$	Condensing design parameter.
$diag$	Diagonalization of the elements in a matrix.
e	Error (setpoint – current value).
$e(k+1)$	Projected error vector.
$E(k)$	Output tracking error.
$E^*(k)$	Coefficient vector while expressing $E(k)$ with wavelet basis.
$Ediag$	Element-by-element diagonalization.
$f(t)$	Function or a signal.
$\boldsymbol{F}$	Weighting matrix of MSP.
g	High-pass filter coefficients.
$\mathbf{G}$	High-pass filtering followed by decimation.
$\mathbf{G}^*$	Adjoint of operator $\mathbf{G}$.
h	Low-pass filter coefficients.
$\mathbf{H}$	Low-pass filtering followed by decimation.
$\mathbf{H}^*$	Adjoint of operator $\mathbf{H}$
$\mathbf{I}$	Identity matrix.
$\boldsymbol{J}$	Performance Index.
K_c	Proportional gain of the system
L^2	Space of square integrable functions.
M	Number of vanishing moments.
$\hat{M}$	A natural number.
m_i	Process input measured at i^{th} sampling instant.
N	Field of natural numbers.
P_B	Blocking projection matrix.
P_C	Condensing projection matrix.
$\hat{P}$	A natural number.
$\mathbf{Q}$	Positive definite weighting matrix.

R	Field of real numbers.
s	Vector of smooth components.
s	Scaling or dilation parameter.
t	Time.
t_0	Centre of Heisenberg's cell along the time axis.
U_c	Continuous future input vector.
U	Vector representation of U_c.
U^*	Unitary transformation of U from pulse basis to wavelet basis.
u	Manipulated variable in the process.
V_j	Closed subspace of $L^2(R)$ at scale j.
W_j	Orthogonal complement of V_j in V_{j-1}.
$\underline{\underline{W}}$	Wavelet matrix.
W_y	An orthogonal matrix.
$w(k+1)$	The vector of the collective effect of un-modeled disturbances.
x	Discrete signal.
x^*	Lost information while approximating x by a lower resolution signal.
$\tilde{x}$	Coefficient vector for pulse basis.
$\tilde{x}^*$	Coefficient vector for wavelet basis.
$\hat{x}_n$	Predicted value of the output at n^{th} instant
$\underline{x}$	Vector of the measured outputs
x_i	Process output measured at i^{th} sampling instant
$y(\mathrm{k})$	Output of the system at k^{th} instant
y	Process output from the model
$\hat{y}^0(k)$	Vector of output variable in absence of further control action.
$\hat{y}(k+1)$	Predicted output vector.
$\mathrm{y}^*(k+1)$	Desired or reference tracjectory of the output.
$\mathbf{Z}$	Field of Integers.

Greek Notations

$\phi(t)$	Scaling function.
$\phi_{m,n}(t)$	Scaling function at scale m and translation n.
Δt	Spread of wavelet in time.
$\Delta\omega$	Spread of wavelet in frequency (bandwidth).
ω	Frequency.
ω_0	Centre of Heisenberg's cell along the frequency axis.
$\psi(t)$	Wavelet.
$\psi_{m,n}(t)$	Wavelet at scale m and translation n.
$\hat{\psi}(\omega)$	Fourier transform of the wavelet.
ε_n	Error between the measured value of the output at n^{th} instant
$\underline{\theta}$	Vector of the unknown parameters
$\Delta u(k)$	Change in the input at k^{th} instant
Π	Transformation matrix
δ	Design parameter
τ	Translation parameter.
τ_I	Integral time constant (sec)
τ_D	Derivative time constant (sec)

Special Notations

$\langle \ldots, \ldots \rangle$	Inner product (Dot product).
$\lvert \ldots \rvert$	Absolute value.
$*$	Convolution.
$\in$	Belongs to.
$\ni$	Is contained in.
$\subset$	Subset of.

$\cup$	Union of.
$\cap$	Intersection of.
$\perp$	Orthogonal to.
$\Leftrightarrow$	Implies and is implied
$\oplus$	Direct sum.
$\|\ldots\|$	Norm
$\supset$	Superset of
$\forall$	For all

Subscripts

$s\ (m \text{ or } j)$	Dilation parameter.
$\tau(n \text{ or } k)$	Translation parameter.
n_u	The number of elements in the vector $U^*(k)$.
n_y	The number of elements in the vector $E^*(k)$.

Superscripts

T	Transpose of a matrix
$*$	Reference trajectory value
$\wedge$	Predicted value
0	Absence of further control action

CHAPTER 1

INTRODUCTION

During the last few decades there has been a dramatic change in the chemical process industries. Industrial processes are now highly integrated with respect to energy and material flow, constrained ever more tightly by high quality product specifications, and subject to increasingly strict safety and environmental emission regulations. These stringent operating conditions often place new constraints on the operating flexibility of the process. Most of the processes are designed to yield an optimum level of performance dictated by safety, productivity, product quality, environmental regulations and profitability. In order to ensure the high quality performance that should be maintained in presence of ever changing environmental influences, an efficient control tool is essential to design. Such a system continuously monitors the performance of the process and executes a suitable corrective action whenever external disturbances cause the process to deviate from the optimum operation. The control system sometimes may also be required to ensure the stability of the closed-loop process. All of these factors produce large economic incentives for reliable, high performance control systems in modern industrial plants.

According to a workshop report on process measurement and control by National Institute of Standards and Technology (www.chemicalvision2020.org/pdfs/workshop_processmanagement.pdf) the chemical industry is among the most successful industries in the United States. The two key enabling technologies for its continued success are process control and chemical measurement that is needed to produce good control actions.

A worldwide survey (Chemical Industry Research Priorities, 1995, ISBN-1-85897-028-8) has put the net worldwide sales volume of process industries at US $1250 billion as of 1993. Assessments made in the same literature shows that, deployment of state of art process control technologies in the plant would potentially bring about improvements in the profit of the order of 5 to 10% of the total sales volume. This shows the importance of controllers in process industries.

The conventional controllers (Proportional or P, Proportional-Integral or PI, and Proportional-Integral-Derivative or PID) give poor performances in nonlinear processes. Most of the chemical processes are nonlinear and in such cases these controllers are not likely to give effective and precise control action.

Fuzzy Logic (FL) and Artificial Neural Network (ANN) based controllers are two independent control schemes, which can provide better control even if an accurate model of the process is not available. But they have their own limitations. In case of FL controllers, it is difficult to predict the closed loop response with analytical tools and simulation is often the only course of evaluation. Also it is difficult to program FL control in machine language (Chidambaram, 2002). ANN based controllers require excessive training of the network, and with that, if the training is not proper, it can give absurd results. ANN controllers often use linear approximations of non-linear control problems using traditional artificial intelligence (AI) approaches (Dracopoulos, 1997). These models are likely to create process/model mismatch and consequently, the model-based controller performance degrade drastically (Haykin, 1998).

A successful alternative to the conventional and model-based controllers are the Model Predictive Controllers (MPCs). There are numerous variations of MPC schemes and of them the Dynamic Matrix Controller (DMC) is the one which is popularly used in the chemical industries.

Recently the process control engineers have been attracted towards wavelet-based analysis to exploit the excellent time-frequency localization and multiresolution approximation capabilities of the wavelets. Researchers (Lee *et al.*, 1992; Palavajjhala *et al.*, 1994; Feng *et al.*, 1996) show through simulation studies that wavelet-based online identification and wavelet-based blocking and condensing techniques can improve the performance of DMC. But, the experimental implementation of the wavelet-based techniques on real pilot plant processes was rarely found even if an extensive literature survey has been conducted.

This work is undertaken to apply wavelet-based identification and control techniques in two pilot plant processes, namely the heat exchanger process and the liquid level process and also to compare the performance of wavelet-based identification and control with those of the conventional ones.

1.1 Objectives of the Work

The present work has been undertaken with the following objectives:

- Identification of heat exchanger process and liquid level process through classical least-square identification and wavelet-based least-square identification methods.
- Comparison of wavelet-based least-square identification with classical least-square identification for the heat exchanger- and liquid level processes.
- Development of the conventional PID controllers and their implementation in the heat exchanger- and liquid level processes for online control of temperature and liquid level, respectively.
- Development of dynamic matrix controllers and their implementation in the heat exchanger- and liquid level processes for online control of temperature and liquid level, respectively.
- Development of wavelet-based dynamic matrix controllers and their implementation in the heat exchanger- and liquid level processes for online control of temperature and liquid level, respectively.
- Comparison of the performance of the wavelet-based dynamic matrix controllers with that of the conventional PID controllers and dynamic matrix controllers in case of both heat exchanger- and liquid level processes.
- Comparison of the performance of the wavelet-based dynamic matrix controllers with that of the Fuzzy Logic and ANN controllers developed by Dasgupta (1999) in case of the same heat exchanger- and liquid level processes for identical conditions.

1.2 Organization of the Thesis

The thesis comprises of seven chapters. The chapters are organized as follows:

Chapter 1 presents a brief introduction of the research work. It provides the motivation for the work and outlines the objectives and presents the organization of the thesis. Chapter 2 includes extensive literature survey on application of wavelet transform in different aspects of advanced control systems. Chapter 3 incorporates a brief introduction to the wavelets and the wavelet transforms. Only the necessary and relevant theories are presented. Chapter 4 gives the description of the experimental setups, namely, the Heat

Exchanger Unit and the Liquid Level Unit along with their accessories and photographs. The application of classical least-square identification and wavelet-based least-square identification for identification of the heat exchanger process and the liquid level process are presented in Chapter 5 along with the relevant theories behind them. Also the comparison between the wavelet-based least-square identification and the classical least-square identification have been presented in this chapter. The implementation of the digital PID control, dynamic matrix control and the wavelet-based dynamic matrix control schemes on the heat exchanger- and the liquid level processes have been presented in Chapter 6 along with the relevant theories. The comparison of the performance of the wavelet-based dynamic matrix controller with those of conventional PID and dynamic matrix controllers are presented in this chapter. Also the performance of the wavelet-based dynamic matrix controller is compared with those of Fuzzy Logic and ANN controllers developed by Dasgupta (1999) for the same experimental setups and for identical conditions. In Chapter 7 the overall conclusions of the work have been stated and the direction for future scope of the research work has been reported.

A comprehensive list of references is given at the end of Chapter 7. Appendix I contains the commuter programs used for the present work. Appendix II contains some selected experimental data obtained from this work.

CHAPTER 2

LITERATURE SURVEY

Over the past few years, wavelets and wavelet-based analysis have found their way into many different fields of science and engineering and has attracted growing attention from mathematicians as well as engineers. Recently the reliability, safety, productivity and quality control in chemical industry have dramatically increased. More effort is being placed on improving the description and analysis of complex phenomena, from fundamental laboratory studies to industrial applications. More accurate and powerful techniques are required to fulfill this goal. Using traditional mathematical tools, one can study the properties of a phenomenon either in time domain or frequency domain. Although the Fourier transform and its inverse allow a passage from one domain to another domain, it does not give a simultaneous view of the phenomenon in both domains. Wavelets are new family of basis functions, which are localized in both time and frequency domains. The wavelet transform is a tool that provides descriptions of functions or signals in the time-frequency plane (Daubechies, 1992). A basis, which preserves the time-frequency information of the signal, can be of significant advantage in control system synthesis and analysis. Wavelets, because of their time-frequency localization and multi-resolution properties, offer such an efficient framework for representation and characterization of signals, especially those non-stationary in nature. Wavelets can also be useful in obtaining process models that describe the process behaviour on different time-scales. These properties are very attractive from process identification and controller design perspective and have been recently shown to be of significant advantage for some of the process engineering problems (Bakshi and Stephanopolous, 1993 & 1994; Palavajjhala et. al., 1994).

2.1 Process Control Prospective of Wavelets

The most appealing properties of wavelets are their time-frequency localization and multiresolution approximation capabilities. Projections of functions or signals onto the wavelet bases result in decompositions, which provide time-frequency information together. These decompositions along with the attractive computational properties of wavelets are exploited in process control applications. An extensive review of applications of wavelets in different aspects of the advanced process control is conducted under the following headings.

2.1.1 *Model Predictive Control*

Model Predictive Control (MPC) has become one of the popular advanced control techniques in the chemical process industries. It refers to a class of control algorithms in which a dynamic process model is used to predict and optimize the plant performance. It provides a systematic framework to solve multivariable control problem, with large interactions and constraints by replacing fixed parameter control laws with on-line dynamic optimization. Of the numerous variations of the MPC algorithm, dynamic matrix control (DMC) is the most widely used in the chemical process industries. Rawlings and Muske (1993) as well as Lee and Yu (1994) indicated that for a robust control, the sampling rate should be fast and the prediction/control horizons should be large. This combination leads to very demanding on-line optimization problem. The solution of such optimization problem can be very time consuming, as it must be solved on-line at each sampling period to generate the control moves. The computational complexity in DMC is reduced by applying techniques called "blocking" and "condensing" (Lee *et al.*, 1992; Palavajjhala *et al.*, 1994). Lee *et al.* (1992) have applied wavelet transform to improve the computational efficiency of infinite horizon multi-scale problems. Through simulation results they have shown that blocking and condensing techniques are conveniently applied after expressing the DMC objective and constraint equations in terms of wavelet bases. Palavajjhala *et al.* (1994) introduced blocking and condensing interval design methodology using information theory and sensitivity analysis. They developed a design procedure for blocking and condensing using Haar wavelet transformation and applied to a shell standard process control problem through simulation study.

Feng *et al.* (1996) proposed a novel approach, model predictive control with simultaneous identification (MPCI) using wavelets, that relies on augmenting the standard on-line MPC optimization with a series of persistent excitation (PE) constraints that current and future process inputs must satisfy over a finite horizon.

2.1.2 *Intelligent Control*

Over the past few years, ANNs have established themselves as a powerful paradigm for modeling complex and nonlinear systems and solving machine learning and pattern recognition problems. Some of the disadvantages of the neural network based system modeling have been the significant computational effort needed in training and adaptation of the networks and the lack of physical meaning of the resulting models. To a large extent, the computational complexity of the networks has been due to the global nature of the node activation functions, which, in turn, led to slow adaptation characteristics. Based on the wavelet transform theory, a new notion of ANN called Wavelet Network (Wave-Net) is proposed by Zhang and Benveniste (1992) and Bakshi and Stephanopoulos (1993). A Wave-Net is an ANN with one hidden layer of nodes, whose basis functions are drawn from a family of orthonormal wavelets. As Neural Network, Wavelet Network also found its use in process control. Gu and Hu (2002) used wave-net for predictive control of a car-like mobile robot. Other uses of wave-net include nonlinear system estimation and control (Cannon and Slotine, 1995; Kavchak and Budman, 1999), process density control (Abu-Zahra and Seth, 2002), robust position control (Shieh *et al.*, 2002), adaptive control of a Continuous Stirred Tank Reactor (CSTR) (Knapp *et al.*, 2001), model formulation (Safavi *et al.*, 1999), bioprocess modeling (Chen and Woodley, 2002), modelling and optimisation of an experimental distillation column (Safavi and Romagnoli, 1997), intelligent control of induction servo motor drive (Wai and Chang, 2002), autonomous rover control (Huntsberger and Rose, 1998), load frequency control in power systems (Hemeida, 2005; Oysal *et al.*, 2005) etc.

2.1.3 *Robust Control*

Models of industrial processes are only approximate representations of actual systems. Therefore, process control designs should accommodate parameter uncertainties and variations. The concept of robustness refers to the preservation of closed-loop stability under allowable variations in system parameters. The objective of robust control theory is to establish design methods that guarantee robustness.

Most of the robust control theory is developed in frequency domain whereas the controller design schemes such as MPC are most attractive when formulated in time domain. Time domain formulations are less amenable to convenient and explicit treatment of the model uncertainty issues. Thus, there is a need for a framework, which can address the time-frequency considerations simultaneously. The wavelet decompositions facilitate such joint time-frequency interpretations of process signals. These decompositions, in turn, are used in formulating new design criteria for controller design, incorporating both time and frequency domain considerations. Bokor (1998) used wavelet bases for approximate identification and robust control. Binder *et al.* (2000) demonstrated that the wavelet based adaptive parameterization is more efficient and robust compared with a uniform parameterization of comparable accuracy. Shieh *et al.* (2002) used wavelet neural network (WNN) for robust position control.

2.1.4 *Adaptive Control*

Process control problems inevitably require on-line tuning of the controller settings to achieve a satisfactory degree of control. If the process operating conditions or the environmental parameters change significantly, the controller may then have to be retuned. If these changes occur frequently, then adaptive control techniques should be considered. An adaptive control system is one in which the controller parameters are adjusted automatically to compensate for changing process conditions.

Most of the chemical engineering systems exhibit behaviours on different time scales. Explicit consideration of these behaviours in adaptation strategies is desirable and wavelets with multiresolution capabilities seem to be the right framework to reformulate the adaptation problems. For instance, the data retention in adaptation, to a large extent is still an unresolved issue. Addressing the problem in time-frequency plane appears to be more natural and convenient than addressing in either domain alone. Potential use of wavelets in adaptive control is addressed by many researchers (Cannon and Slotine, 1995; Binder *et al.*, 2000; Luo *et al.*, 2003). Knapp *et al.* (2001) used an adaptive control algorithm with a wavelet neural network model for control of a Continuous Stirred Tank Reactor (CSTR). Kavchak and Budman (1999) used adaptive neural network structures for nonlinear process estimation and control. Wai and Chang (2002) proposed an intelligent control system that is on-line trained WNN controller with adaptive learning rate to control the rotor position of an induction servo motor drive.

2.1.5 *Nonlinear Control*

Most physical processes exhibit nonlinear bahaviour to some degree. However, linear control techniques such as conventional PID control are still very effective if the nonlinearities are rather mild or a highly nonlinear process operates over a narrow range of conditions. For some highly nonlinear processes, the second condition is not satisfied and as a result, linear control strategies may not be adequate. For these situations, nonlinear control strategies can provide significant improvements over PID control.

Solution of the nonlinear ordinary differential equations (ODEs) and partial differential equations (PDEs) using wavelet basis functions has been the focus of many researchers (Liandrat *et al.*, 1992; Nikolaou and You, 1992 and 1994; Adrover *et al.*, 2000; Liu at al., 2000; Mahadevan and Hoo, 2000; Cruz *et al.*, 2001 & 2002; Bindal *et al.*, 2003; Lepik, 2005). Multiresolution approximation properties of wavelets are of advantage in localizing the approximation bias to desired regions of solution e.g. later parts of prediction horizon in nonlinear MPC. This has significant relevance for reducing the on-line computational requirements of nonlinear MPC implementations.

Shmilovici and Oded Maimon (1996 & 1998) demonstrate that fuzzy spline wavelets can be used for efficient solution of various types of differential equations. The advantage in using fuzzy spline wavelets for the solution of differential equations is that the solution would enjoy the excellent numerical and computational characteristics of the fast wavelet transform, while retaining the explanatory power of fuzzy system. They have demonstrated the method with the feedforward control of a flexible robotic arm. Poterasu (2001) solves a nonlinear control problem of multibody systems using wavelet transform. He presents a wavelet collocation method to solve a nonlinear time-evolution problem. Razzaghi and Yousefi (2001) have presented a numerical technique for solving the nonlinear problems of the calculus of variations using Legendre wavelets method. They claim that Legendre wavelets method is general, easy to implement, and yields very accurate results. They have formulated the brachistochrone problem as a nonlinear optimal control problem. Liu and Cameron (2001) have proposed a new wavelet-based method for solving population balance equations with simultaneous nucleation, growth and agglomeration. This technique is very general, yet powerful and overcomes the crucial problems of numerical diffusion and stability that often characterize previous techniques in this area.

2.1.6 *Process Modeling and Control*

Dynamic process models play a central role in the subject of process dynamics and control. For model-based control strategies, the process model is a part of the control law. Safavi *et al.* (1999) employed wavelet-based neural networks to simplify the model of a distillation column. The simplified model preserves the accuracy of the model together with the availability of the required internal variables of the model. Similar applications (Chen and Woodley, 2002; Knapp *et al.*, 2001) are found in the literature.

Mathematical models that describe distributed parameter systems are generally composed of systems of partial differential and algebraic equations. The methods that solve these systems usually yield a high-order solution. However, for controller synthesis and practical considerations, a low-order model is preferred. Some researchers (Mahadevan and Hoo, 2000; Briesen and Marquardt, 2000; Adrover *et al.*, 2000; Zhao *et al.*, 2005) address the development of model reduction through wavelet-based multiresolution methods that yield a low order model for controller design.

2.1.7 *Process Identification and Control*

For the cases when the theoretical modeling is impossible, impractical or inconvenient, the useful model of the system can be obtained by process identification. Process identification involves constructing a process model strictly from experimentally obtained input-output data, with no recourse to any laws concerning the fundamental nature and properties of the system (Ogunnaike and Ray, 1994).

Most of the system identification techniques can be represented as projection onto a basis set (Carrier and Stephanopoulos, 1992). The properties of this basis set, to a large extent, influence the range of validity of the identified model. Stephanopoulos and Carrier (1991) have proposed control relevant model identification methodologies based on the multiresolution analysis of the input-output data. They have used the orthonormal wavelet framework to construct reduced order models which are accurate in a given time-frequency range. They have also utilized the time-frequency properties of wavelets in quantifying the associated model uncertainty and calculated error bounds for robust controller design.

The identification experiments often result in input-output data with small signal-to-noise ratio. This results in inaccurate model parameter estimates. Prefilters are used to separate useful information from the noise in the input-output data and to improve parameter estimates. Palavajjhala *et al.* (1996) have presented a systematic design procedure for selecting a prefilter using discrete wavelet transforms. Their design procedure provides explicit information on the compromises in prefilter design. They have applied the prefilter design procedure to identify a second-order output model. Once the model of a process is identified, it can be used for the design of the process controller. The process controller design requires that the information from the process be extracted in pieces that are localized in both time and frequency. Such an extraction process allows the separation of valuable signal information from the effects of nonstationary disturbances and noise. The wavelet transform provides an efficient approach for such decomposition. Carrier and Stephanopoulos (1998) have used the method of modulating functions in conjunction with the wavelet decomposition and have demonstrated that recursive state-space models, which are multiple in character and suitable for the design of model predictive controllers, could be readily constructed with lower levels of modeling errors than yielded by traditional techniques. Their method is especially suitable for the identification of time-varying and nonlinear models. Wang *et al.* (2000) have proposed a wavelet-based robust M-estimation method for the identification of nonlinear systems. M-estimator takes a flexible, nonparametric approach and has the advantage of directly estimating the error distribution from the data. It is optimal over any error distribution in the sense of maximum likelihood estimation. They have used a Monte-Carlo study on a nonlinear chemical engineering example to compare the results with various previously utilized methods. Coca and Billings (2001) have introduced a wavelet-based methodology for identifying nonlinear autoregressive moving average exogenous (NARMAX) models from noise-corrupted data. They have inferred model complexity from the data while reducing the computational costs by introducing an adaptive model sequencing strategy. They have used this in conjunction with an iterative orthogonal-forward-regression routine coupled with model validity tests to identify sparse but accurate wavelet series representations of non-linear processes. Experimental data from two real systems, a liquid level system and from a civil engineering structure have been used to illustrate the effectiveness of the wavelet-based identification procedure.

The problem of system identification using wavelet basis has been addressed by several researchers (Tsatsanis and Giamakis, 1993; Schoenwald, 1993; Pati *et al.*, 1993; Xin and Sano, 1995; Feng *et al.*, 1996; Yang *et al.*, 1997; Kosanovich *et al.*, 1997; Nikolaou and Vuthandam, 1998; Antonopoulos-Domis and Tambouratzis, 1998; Bokor, 1998; Takahashi *et al.*, 1998; Doroslovacki *et al.*, 1998; Kaufhold *et al.*, 1999; Ghanem and Romeo, 2000; Piombo *et al.*, 2000; Wang *et al.*, 2001; Kijewski and Kareem, 2002; Fadili and Bullmore, 2002; Pislaru *et al.*, 2003; Alvin *et al.*, 2003; Le and Argoul, 2004; Al-Assaf *et al.*, 2004; Dorfan *et al.*, 2004; Sun *et al.*, 2004; Mallinson *et al.*, 2004). Carrier and Stephanopoulos (1998) have shown that the localized time-frequency nature of wavelet provides superior performance over the standard techniques for time-varying systems as well as systems corrupted by noise and disturbance. Feng *et al.* (1996) have used wavelet basis as the modulation function for system identification.

Wavelet offers a powerful framework for nonlinear system identification (Sjöberg *et al.*, 1995; Juditsky *et al.*, 1995; Staszewski, 1998; Hasiewicz and Pawlak, 2000; Ghanem and Romeo, 2001). Kosanovich *et al.* (1995) have introduced a novel wavelet transform (called Poisson wavelet transform) and have applied it for system identification, parameter estimation and model validation. Sjöberg *et al.* (1995) have described different approaches, such as neural networks, radial basis networks, wavelet networks and hinging hyperplanes, as well as wavelet-transform-based methods, and models based on fuzzy sets and fuzzy rules for system identification in a common framework. They have discussed about the choices that have to be made and the considerations that are relevant for a successful system identification application of these techniques.

Srivastava *et al.* (2005) have proposed two new fuzzy wavelet neural networks (FWNNs) by utilizing some of the important properties of wavelets like denoising, compression, multiresolution along with the concepts of fuzzy logic and neural network for identifying a non-linear system. They have simulated a control system for maintaining the output at a desired level by using the identified models. They have also designed self-learning fuzzy neural network (FNN) controller and showed through simulation results that the controller was adaptive and robust.

2.2 Justification of the Work

As can be seen from the literature review, the wavelet-based process identification has now been established with sufficient maturity, but limited progress has been achieved on wavelet-based control. Lee *et al.* (1992) have been the pioneers in applying wavelet transform to improve the computational efficiency of MPC. Through simulation studies they have proposed an infinite prediction horizon-based MPC algorithm that has desirable stability and constraint-handling properties. They achieved closed-loop stability and reliable constraint-handling properties by pushing the prediction horizon out to infinity. They have shown that an equivalent finite horizon problem can be formulated, but can potentially lead to large online computational requirement. They have demonstrated that a significant reduction in the computational complexity can be achieved without sacrifice in performance via techniques called blocking and condensing. They have also shown that the blocking and condensing techniques are conveniently applied after expressing the MPC objective and constraint equations in terms of wavelet bases. But their work was a simulation study. Later on Palavajjhala *et al.* (1994) extended the work of Lee *et al.* (1992). They have developed a systematic design procedure for blocking and condensing in quadratic DMC using Haar wavelet from information theory and sensitive analysis. They have applied the design procedure to Shell process control problem through simulation study. Feng *et al.* (1996) used wavelets for model predictive control with simultaneous identification (MPCI). They have provided a systematic way to design the process input for process identification under feedback control keeping the control performance deterioration at a minimum level. Their work was also based on simulation studies.

It is observed from the literature survey that limited progress has been made for investigating the wavelet-based techniques on real time processes. The main endeavour of this thesis, therefore, is to carry out extensive experimentation on the frequently encountered processes, such as, heat exchanger- and liquid level processes. The experimental input-output results are used in the identification and development of wavelet-based controllers. The wavelet-based controllers are then used online for controlling temperature and liquid level in the heat exchanger and the liquid level unit, respectively. The closed loop responses using wavelet-based techniques for the above-mentioned processes are also compared with those obtained using conventional and advanced controllers to evaluate their applicability.

The comparison between the performances of the wavelet-based dynamic matrix controller and the conventional dynamic matrix controller was not found in the literature. It is of worth a study to see the performances of both the controllers in real pilot plant processes.

The conventional PID controllers have been extensively used in process industries. Therefore, it is important to compare the performances of the wavelet-based controller and the conventional PID controllers.

The present work is an attempt to bridge the above-mentioned research gaps.

CHAPTER 3

INTRODUCTION TO WAVELETS

In this chapter, a brief overview on wavelet and wavelet transform is presented. Basically, the discussion is on the definition and properties of wavelets, their time-frequency localization, discretization of scaling and translation parameters, multiresolution analysis, scaling function and its two scale relations, continuous and discrete wavelet transforms, reconstruction of the signal using fast wavelet algorithm and wavelet packet algorithm.

3.1 Introduction

A *wave* is usually defined as an oscillating function of time or space and is periodic. In contrast, a *wavelet* is a "small wave", which is localized in time and frequency (Burrus *et al.*, 1998). Figure 3.1 demonstrates a wave and a wavelet.

The theoretical concept of wavelet was first presented by the French geophysicist Jean Morlet in 1980s (Morlet *et al.*, 1982; Goupillaud *et al.*, 1984a,b) as a tool for signal analysis in view of applications for the analysis of seismic data (Temme, 1993). He subsequently collaborated with the theoretical physicist Alex Grossmann (Grossmann and Morlet, 1984; Grossmann *et al.*, 1985, 1986 & 1987), where wavelets were broadly defined in the context of quantum physics. This collaboration provided an understanding of the wavelets based on physical intuition. But the mathematical theory of wavelets can be traced much further back in time. In many ways it was initiated by Joseph Fourier in 1807 with his theories of frequency analysis. Subsequently the first mention of mathematical objects that are wavelets appeared in an Appendix of the Ph.D. thesis of Haar in 1909. The mathematical study of the discrete wavelets was started with the introduction of "frame" by Daubechies, Grossmann and Meyer (Daubechies *et al.*, 1986; Daubechies and Grossmann, 1988; Hernández and Weiss, 1996). Also many researchers tried to find the orthonormal bases of wavelets (Stromberg, 1981; Battle, 1987; Meyer, 1988). In 1988 Daubechies provided a major breakthrough by constructing families of orthonormal wavelets with compact support (Daubechies, 1988). She was inspired by the work of Mallat and Meyer in the field of multiresolution analysis and by Mallat's algorithms (Mallat, 1988 & 1989) in which he used this analysis for decomposition and reconstruction of images (Daubechies, 1990 & 1991; Daubechies *et al.*, 1992). After that wavelet theory has attracted a lot of attention in many fields, from theoretical study to applications (Daubechies, 1992 &1993). The popularity of wavelet transform is growing because of its ability to reduce distortion in the reconstructed signal while retaining all the significant features present in the signal (Sripathi, 2003).

3.2 Wavelets

Wavelets are the building blocks of wavelet transforms, just as trigonometric functions of different frequencies are the building blocks used in Fourier transforms. The name "wavelet" has been first suggested by Yves Meyer and Jean Morlet (Jawerth and Sweldens, 1993). The wavelets are generated by the dilation and the translation of a single prototype function called "mother wavelet" (Graps, 1995; Bindal *et al.*, 2003). The mother wavelet is an absolutely integrable function and is denoted by $\psi(t)$. A family of wavelets $\{\psi_{s,\tau}(t)\}$ are obtained from the mother wavelet as:

$$\psi_{s,\tau}(t) = \frac{1}{\sqrt{|s|}}\psi\left(\frac{t-\tau}{s}\right) \qquad s,\tau \in R \quad and \quad s \neq 0 \tag{3.1}$$

where s is called dilation (scaling) parameter, and τ is the translation parameter. The factor $1/\sqrt{|s|}$ is used to ensure that the energy of the dilated and translated versions are equal to that of the mother wavelet. For the wavelets to be useful analysing (or basis) functions, the mother wavelet must satisfy certain desirable conditions. The condition for a function $\psi(t)$, with Fourier transform $\hat{\psi}(\omega)$, to be a wavelet is

$$C_{\psi} = \int_{-\infty}^{\infty} \frac{|\hat{\psi}(\omega)|^2}{|\omega|} d\omega < \infty . \tag{3.2}$$

This condition is called the "admissibility" condition (Grossmann *et al.*, 1985; Daubechies, 1992). If the mother wavelet $\psi(t)$ is an absolutely square integrable function ($\psi(t) \in L^2(R)$), as is usually the case, then its Fourier transform $\hat{\psi}(\omega)$ is continuous (Palavajjhala *et al.*, 1994). If $\hat{\psi}(\omega)$ is continuous, C_{ψ} can be finite only if

$$\hat{\psi}(0) = 0 \Leftrightarrow \int_{-\infty}^{\infty} \psi(t)dt = 0 . \tag{3.3}$$

A wavelet must therefore be an oscillatory function with zero mean (symmetric with respect to time axis). Other desirable properties for a function to be a useful wavelet are, smoothness ($\psi(t)$ should be differentiable n times with continuous derivatives), good time localization ($\psi(t)$ and its derivatives should decay rapidly), good frequency localization ($\hat{\psi}(\omega)$ should decay sufficiently fast as frequency $\omega \rightarrow \infty$ and $\hat{\psi}(\omega)$ should be sufficiently flat near $\omega = 0$) and a large number of vanishing moments (number of moments of the wavelet that are zeros). A wavelet is said to have M vanishing moments if

$$\int_{-\infty}^{\infty} t^l \psi(t) dt = 0 \qquad for \quad l = 0,1,\ldots,\mathrm{M}. \tag{3.4}$$

3.2.1 *Time-frequency localization*

Associated with each wavelet, there is a time localization and a frequency localization. A measure of where $\psi(t)$ is centred along the time axis is given by (Rao and Bopardikar, 1998)

$$t_0 = \int_{-\infty}^{\infty} t|\psi(t)|^2 dt. \tag{3.5}$$

Similarly, the measure of where $\hat{\psi}(\omega)$ is centred along the frequency axis is given by

$$\omega_0 = \int_{-\infty}^{\infty} \omega|\hat{\psi}(\omega)|^2 d\omega. \tag{3.6}$$

A measure of duration of the wavelet or the spread in time is given by (Akansu and Haddad, 1992; Papoulis, 1977)

$$\Delta t = \sqrt{\int_{-\infty}^{\infty} (t - t_0)^2 |\psi(t)|^2 dt}. \tag{3.7}$$

The spread of the wavelet in frequency or bandwidth is given similarly by

$$\Delta\omega = \sqrt{\int_{-\infty}^{\infty} (\omega - \omega_0)^2 |\hat{\psi}(\omega)|^2 d\omega}. \tag{3.8}$$

The wavelet $\psi(t)$ is centred on (t_0, ω_0) in the time-frequency plane and contains information from the time interval $(t_0 - \Delta t \;,\; t_0 + \Delta t)$ and the frequency interval

$(\omega_0 - \Delta\omega \quad , \quad \omega_0 + \Delta\omega)$. The two intervals define a time-frequency cell (also called as Heisenberg's cell), whose area is restricted by the uncertainty principle, i.e. $\Delta t \times \Delta\omega \geq \frac{1}{2}$. The time localization and frequency localization remain unchanged with translation of the wavelet. However, with dilation (or scaling), the time and frequency localizations change as follows:

$$\Delta t_s = s\Delta t \quad \text{and} \quad \Delta\omega_s = \Delta\omega / s \tag{3.9}$$

Thus, the wavelets have good time localization for small values of s, which corresponds to high frequency. For large values of s, the wavelets have good frequency localization.

3.2.2 *Discrete Wavelets*

Wavelets are broadly classified into "continuous wavelets" and "discrete wavelets". The dilation parameter s and the translation parameter τ vary continuously in the case of continuous wavelets, but are restricted to a discrete lattice in the case of discrete wavelets. The parameters s and τ are discretised as

$$s = s_0{}^m , \qquad = n\ {}_0 s_0{}^m , \quad m,n \in Z. \tag{3.10}$$

Typically, $s_0 = 2$ and $\tau_0 = 1$ is used. Substituting $s = 2^m$ and $\tau = n2^m$ in Equation (3.1) we get the discrete wavelet as

$$\psi_{m,n}(t) = 2^{-m/2}\psi(2^{-m}t - n). \tag{3.11}$$

where m is the dilation or scaling parameter and n is the translation parameter.

3.2.3 *Multiresolution Analysis*

Resolution means "information content". The concept of multiresolution can be understood in the following way:

When a person goes away from the Earth's surface by an airplane, he gradually looses the detail information of the landscape. Actually he observes the landscape at different resolutions at different times. Initially he sees everything visible at the earth surface. It can be called as a *fine* or high-level resolution of the landscape. As he goes up, he cannot see humans and animals, but can see houses and cars. Next he cannot see houses and cars but can see only the major features like mountains, rivers and larger geological structures. Each stage of observation is a *coarse* or low-level resolution of the landscape compared to the previous one. Mathematically, each stage of resolution is called a *space*, which can be represented as a linear combination of some suitable basis (Just like any point in an Euclidean space can be represented by the linear combination of its basis set {(1,0,0), (0,1,0) (0,0,1)}). Each resolution space is a subset of the previous resolution space of higher resolution. So we can say that "human-animal space" is a superset of "house-car space", which, in tern, is a superset of "mountain-river space". Calling each subspace V_j, the nested subspaces can be written as

$$\cdots V_{-2} \supset V_{-1} \supset V_0 \supset V_1 \supset V_2 \cdots \tag{3.12}$$

Multiresolution Analysis (MRA) provides a formal approach to constructing the orthonormal basis. The idea of MRA is to write the discrete signals by beginning with the finest-resolution signal and successively discarding details to create low-resolution versions, ultimately ending with the coarsest resolution. The successive approximations thus correspond to different resolutions, hence the name multiresolution. In MRA, the space of square integrable functions, $L^2(R)$ is decomposed into closed linear vector spaces V_m for all $m \in Z$, with the following properties:

- $V_m \subset V_{m-1}$ - The coarser subspace is contained in the finer subspace.
- $\bigcap_{m \in Z} V_m = \{0\}$ - Separation condition and

  $\bigcup_{m \in Z} V_m = L^2(R)$ - Condition for completeness.
- $f(t) \in V_m \Leftrightarrow f(2t) \in V_{m-1}$ - Scaling property.
- $f(t) \in V_0 \Leftrightarrow f(t+1) \in V_0$ -Shifting property.
- There exists $\phi(t) \in V_0$ such that $\{\phi(t-n)\}_{n \in Z}$ is an orthonormal basis for V_0.

The function $\phi(t)$ is called the "scaling function" of the MRA. This satisfies the condition

$$\int_{-\infty}^{\infty} \phi(t)dt = 1. \tag{3.13}$$

The orthonormal bases for all the subspaces in this structure can be generated using translations and dilations of the scaling function as

$$\phi_{m,n}(t) = 2^{-m/2}\phi(2^{-m}t - n) \qquad m,n \in Z. \tag{3.14}$$

Since $\phi(t) \in V_0 \subset V_{-1} \ni \phi(2t)$, there exists a finite sequence $h(k)$ such that the scaling function $\phi(t)$ can be written as a linear combination of $\phi(2t)$. The scaling function therefore satisfies the *two-scale relation*

$$\phi(t) = \sum_k h(k)\phi(2k-1). \tag{3.15}$$

The finite sequence $\{h(k)\}$ is known as the low-pass filter coefficients, as the frequency response of the scaling function is that of a low-pass filter.

Let W_j be the orthogonal complement of V_j in V_{j-1}. It satisfies

$$V_{j-1} = V_j \oplus W_j \quad \text{and} \quad V_j \perp W_j \tag{3.16}$$

where the symbol $\oplus$ stands for direct sum. It means that the subspace W_j contains the "detail" information needed to go from an approximation at resolution j to an approximation at resolution *j-1*. If $\{\psi(t-k)\}_{k\in Z}$ is an orthonormal basis of W_0, the orthonormal bases for all W_j can be generated by dilations and translations of the mother wavelet as given in equation (3.11).

Since $\psi(t) \in W_0 \subset V_{-1} \ni \phi(2t)$, there exists a finite sequence $g(k)$ such that the scaling function $\psi(t)$ can be written as a linear combination of $\phi(2t)$. The mother wavelet therefore satisfies the *two-scale relation*

$$\psi(t) = \sum_k g(k)\phi(2k-1). \tag{3.17}$$

The finite sequence $\{g(k)\}$ is known as the band-pass filter coefficients, as the frequency response of the wavelet is that of a band-pass filter. Equation (3.17) suggests that the wavelet can be constructed once the scaling function is constructed using equation (3.15). Rather than beginning with a mother wavelet, MRA starts with the scaling function (Kaiser, 1994).

3.2.4 *Wavelet Families*

There are a number of wavelets with different properties available in the literature. Some of the wavelets are the Haar wavelet (Haar, 1910; Burrus *et al.*, 1998), Daubechies' orthonormal wavelets of compact support (Daubechies, 1988; Walker, 1999), Daubechies' Coiflets (Belkin *et al.*, 1991; Daubechies, 1992), Chui-Wang's block-spline semi-orthogonal wavelets (Chui and Wang, 1992), Battle-Lemarie's orthonormal wavelets (Battle, 1987), Lemarié-Meyer wavelets (Hernández and Weiss, 1996), Cohen et al's biorthogonal wavelets (Cohen *et al.*, 1993), Shannon's wavelet (Goswami and Chan, 1999), Meyer's wavelet (Kaiser, 1994), Mexican hat wavelet (Rao and Bopardikar, 1998), Alpert's wavelets (Alpert, 1992), Franklin wavelets (Hernández and Weiss, 1996), Symlet wavelet (Alsberg, internet), Spline wavelets (Chui *et al.*, 1994), Malvar wavelets (Suter and Oxley, 1994), Marr wavelet (Graps, 1995), and Morlet's wavelet (Morlet *et al.*, 1982), etc.

Figure 3.2 illustrates some of the commonly used wavelets. Haar wavelet is one of the oldest and the simplest wavelet. Therefore, any discussion on wavelet starts with the Haar wavelet. It is an orthonormal wavelet defined as:

$$\psi(t) = \begin{cases} 1 & 0 \le t < \frac{1}{2} \\ -1 & \frac{1}{2} \le t < 1 \\ 0 & \text{Otherwise} \end{cases} \tag{3.18}$$

The scaling function for the Haar wavelet is a pulse function defined as:

$$\phi(t) = \begin{cases} 1 & 0 \le t < 1 \\ 0 & \text{Otherwise} \end{cases} \tag{3.19}$$

The Haar scaling function and Haar wavelet are shown in Figure 3.3 and Figure 3.4, respectively.

Daubechies wavelets are the most popular wavelets. They represent the foundations of wavelet signal processing and they are used in numerous applications. They are orthonormal and compactly supported. They are named according to the number of filter coefficients they possess, e.g. Daub4 means "Daubechies wavelet with 4 coefficients". Daubechies wavelets of 4, 6, 8, 10, 12, 16, 20 and 40 coefficients are found in the literature (Burrus *et al.*, 1998). There are no analytical expressions for the Daubechies wavelets and scaling functions. Figures 3.5 and 3.6 show some of the Daubechies scaling functions and wavelets, respectively.

The Symlets and Coiflets are also compactly supported orthogonal wavelets. These wavelets along with Meyer wavelets are capable of perfect reconstruction. The Meyer, Morlet and Mexican Hat wavelets are symmetric in shape. The wavelets are chosen based on their shape and ability to analyse the signal in a particular application.

3.3 Wavelet Transform

Mathematical transformations are applied to signals to obtain further information from those signals that is not readily available in the raw signals (Polikar, 1996a). Wavelet Transform (WT) is a tool that cuts data, functions, or operators, into different frequency components, and then studies each component with a resolution matched to its scale (Daubechies, 1992). It represents a square integrable time function in terms of wavelets (and scaling functions).

Fourier Transform (FT) of a signal tells what frequency components exist in a signal, but it does not give any information regarding when in time these frequency components occur (Polikar, 1996b). So, FT is unsuitable for analysing non-stationary signals (signals whose frequency changes with time). Short-Time Fourier Transform (STFT) is used to overcome this problem. But it suffers from resolution problem, i.e., frequency resolution is poor when STFT uses narrow window and time resolution is poor while it uses wide window. Moreover, it uses a fixed window for analysis of a particular signal.

WT was developed to overcome the shortcomings of the STFT. While STFT gives a constant resolution at all frequencies, the WT uses MRA by which different frequencies are analysed with different resolutions. In WT, as frequency increases, the time resolution increases (frequency resolution decreases); likewise, as frequency decreases, the frequency resolution increases (time resolution decreases). Thus, a certain high frequency component can be located more accurately in time than a low frequency component and a low frequency component can be located more accurately in frequency compared to high frequency component. Time frequency tiling for STFT and WT is shown in Figure 3.7.

3.3.1 *Continuous Wavelet Transform*

Continuous Wavelet Transform (CWT) of a function $f(t) \in L^2(R)$ involves computation of the inner product $\langle f(t), \psi_{s,\tau}(t) \rangle$ for $s, \tau \in R, s \neq 0$. The inner products for various values of parameters s and τ are called the *Continuous Wavelet Coefficients* and are defined as:

$$\langle f(t),\psi_{s,\tau}(t)\rangle = \frac{1}{\sqrt{|s|}}\int_{-\infty}^{\infty} f(t)\psi\left(\frac{t-\tau}{s}\right)dt. \tag{3.20}$$

3.3.2 *Discrete Wavelet Transform*

Discrete Wavelet Transform (DWT) of the function $f(t)$ involves computation of the above inner product at discrete values of dilation and translation parameters. These inner products are the Discrete Wavelet Coefficients and are defined as:

$$\langle f(t),\psi_{m,n}(t)\rangle = 2^{-m/2}\int_{-\infty}^{\infty} f(t)\psi(2^{-m}t-n)dt. \tag{3.21}$$

For the orthogonal case, the reconstruction formula for the function $f(t)\in L^2(R)$ from the discrete wavelet coefficients is

$$f(t)=\sum_{m,n\in Z}\langle f(t),\psi_{m,n}(t)\rangle\ \psi_{m,n}(t). \tag{3.22}$$

The parameters m and n vary between -∞ and ∞. This means that we need an infinite number of wavelets to get the job. Every time we stretch the wavelet in the time domain with a factor of 2, its bandwidth is halved. In other words, with every wavelet stretch we cover only half of the remaining spectrum. So practically we can never cover the whole spectrum. The solution to this problem is simply not to try to cover the spectrum all the way down to zero with wavelet spectra, but to use a cork to plug the hole when it is small enough (Valens, 1999). This cork then is a low-pass spectrum and it belongs to the *scaling function*, which is discussed in MRA.

In practice, signals are measured at a finite resolution, and therefore, the value of m should be restricted between [0,L], where $m=0$ corresponds to the finest scale and $m=L$ corresponds to the coarsest scale. We can stretch the wavelet from scale $m=1$ to scale $m=L$ and use the scaling function at scale $m=L$ to capture the remaining spectra. Thus, we can reconstruct the original signal (signal at scale $m=0$) as

$$f(t)=\sum_{m\in[1,L],n\in Z}\langle f(t),\psi_{m,n}(t)\rangle\ \psi_{m,n}(t)+\langle f(t),\phi_{L,n}(t)\rangle\ \phi_{L,n}(t). \tag{3.23}$$

Equation (3.23) includes two terms on the right hand side. The first term is the projection of $f(t)$ on wavelets at different scales or resolutions. The second term is the projection of the function on the coarsest scaling function. The inner product in the second term is defined as

$$\langle f(t), \phi_{L,n}(t) \rangle = 2^{-L/2} \int_{-\infty}^{\infty} f(t) \phi(2^{-L}t - n) dt. \tag{3.24}$$

Thus, DWT can be interpreted as a set of signals resulting from smoothing of a function $f(t)$ from scale $m = 1$ to scale $m = L$. The term *level* is sometimes used to refer to a scale in this decomposition.

3.4 Mallat's Fast Wavelet Algorithm

Mallat (1989) used a hierarchical decomposition using pyramidal algorithm to compute the wavelet coefficients efficiently. Let, the sequence c_0 denote the projections of the function $f(t)$ on the scaling function at scale $m = 0$ and $n \in Z$. The samples of a continuous function $f(t)$ at the finest scale are often used instead of c_0. The projections of the function $f(t)$ on the scaling function at the next scale can be computed using

$$c_{m,n} = \langle f(t), \phi_{m,n}(t) \rangle = \sum_k h(k - 2n) c_{m-1,k} \tag{3.25}$$

where,

$$h(n) = 2^{-1/2} \int_{-\infty}^{\infty} \phi(t/2) \phi(t - n) dt. \tag{3.26}$$

Thus, the sequence c_m, which is the projection of the function $f(t)$ on the scaling functions $\phi_{m,n}(t)$ for all $n \in Z$, can be computed using

$$c_m = \mathrm{H} c_{m-1} \tag{3.27}$$

Operator H can be interpreted as the convolution of the sequence c_{m-1} with the filter coefficients $h(n)$ followed by decimation (dropping even terms). The projection on the wavelet is computed using

$$d_{m,n} = \langle f(t), \psi_{m,n}(t) \rangle = \sum_k g(k - 2n) c_{m-1,k} \tag{3.28}$$

where,

$$g(n) = 2^{-1/2} \int_{-\infty}^{\infty} \psi(t/2) \phi(t - n) dt. \tag{3.29}$$

The sequence d_m, which is the projection of the function $f(t)$ on the wavelets $\psi_{m,n}(t)$ for all $n \in Z$, can be computed using

$$d_m = \mathrm{G} c_{m-1} \tag{3.30}$$

Operator G can be interpreted as convolution of the sequence c_{m-1} with the filter coefficient $g(n)$ followed by decimation. From equations 3.25 and 3.28, the following recursive wavelet decomposition algorithm is obtained:

$$c_j = \mathrm{H} c_{j-1} \text{ and } d_j = \mathrm{G} c_{j-1} \tag{3.31}$$

Figure 3.8 depicts three stage of this algorithm. The projections on the scaling function are recursively decomposed to obtain coarser approximations. The projections on the wavelets at each scale contain the finer details.

The reconstruction algorithm is similar to the decomposition algorithm, except the steps are now reversed:

$$c_{j-1} = \mathrm{H}^* c_j + \mathrm{G}^* d_j \tag{3.32}$$

Figure 3.9 depicts the reconstruction from the three-stage decomposition shown in Figure 3.8. Finer details captured by the wavelets are added to the coarser approximations to reconstruct the original function. Operators H^* and G^* are the adjoints of operators H and G, respectively. The adjoint operators can be interpreted as upsampling (padding even terms with zeros) followed by convolution with the respective conjugate filter coefficients.

Operators H and G used in the wavelet decomposition are low-pass and band-pass filters (Mallat, 1989), respectively and their filter coefficients are h(n) and g(n), respectively.

3.5 Wavelet Packet Algorithm

Coifman and Wickerhauser (1992) introduce a general class of orthonormal bases by constructing a library of modulated waveforms out of which various bases, including the wavelet basis, can be extracted. They show that this library of orthonormal bases is a binary tree, and they develop a fast algorithm to search for a minimum information cost representation. The library of bases is called the *wavelet packet* and the decomposition of a signal onto these bases is called the *Wavelet Packet Decomposition*.

The idea behind the wavelet packet decomposition is illustrated in Figure 3.10. The projections of a function on the scaling functions and the wavelets are recursively decomposed to obtain a binary tree. The wavelet packet decomposition of a function sampled at eight equally spaced points is presented as follows.

Let the sampled function be represented by the sequence $x=\{x_1, \; x_2, \; \dots, \; x_8\}$. The wavelet packet decomposition of this sequence is obtained by applying filters H and G recursively to form a binary tree shown in Figure 3.11. The root of the tree is a cell consisting of the original sequence x. The filters H and G are applied recursively on the parent cell to yield two child cells.

In Figure 3.11, the left child cell $s=\{s_1, \; s_2, \; s_3, \; s_4\}$and the right child cell $d=\{d_1, \; d_2, \; d_3, \; d_4\}$at Level 1 are obtained by applying the filters H and G, respectively to the sequence x. The sequences s and d for real valued filters $h=\{h_1 \quad h_2 \quad h_3 \quad h_4\}$ and $g=\{g_1 \quad g_2 \quad g_3 \quad g_4\}$are computed as:

$$s=Hx \quad \text{and} \quad d=Gx \tag{3.33}$$

where,

$$H=\begin{bmatrix} h_1 & h_2 & h_3 & h_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_1 & h_2 & h_3 & h_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & h_1 & h_2 & h_3 & h_4 \\ h_3 & h_4 & 0 & 0 & 0 & 0 & h_1 & h_2 \end{bmatrix} \tag{3.33a}$$

$$G=\begin{bmatrix} g_1 & g_2 & g_3 & g_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & g_1 & g_2 & g_3 & g_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & g_1 & g_2 & g_3 & g_4 \\ g_3 & g_4 & 0 & 0 & 0 & 0 & g_1 & g_2 \end{bmatrix} \tag{3.33b}$$

$$x=[x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5 \quad x_6 \quad x_7 \quad x_8]^T \tag{3.33c}$$

$$s=[s_1 \quad s_2 \quad s_3 \quad s_4]^T \tag{3.33d}$$

$$d=[d_1 \quad d_2 \quad d_3 \quad d_4]^T \tag{3.33e}$$

The wavelet packet decomposition shown in Figure 3.11 can be interpreted in terms of approximating sequences. The sequence s can be considered as a coarser approximation of the original sequence x. Let the filters H and G be constructed using orthonormal basis. Let the normalization be done so that the total energy before and after applying these filters to a sequence remains unchanged, i.e. the transformation is loss-less. The energy of a sequence is defined as the sum of the squares of the elements. Therefore, the error in approximating sequence x by sequence s can be measured as the energy of sequence d. Furthermore, the coarser approximation of elements $\{x_1, x_2\}$ is the element s_1 and the error in approximation is energy of element d_1. Carrying this argument further to other levels in the tree, the first element at Level 3 (sss) is the coarsest approximation to the original sequence, and the error (energy loss) associated with this approximation is $d_1^2 + d_2^2 + d_3^2 + d_4^2 + ds_1^2 + ds_2^2 + dss^2$. The element sss along with dss, ds_1, ds_2, d_1, d_2, d_3, d_4 constitute the wavelet representation of the original sequence x. Since the filter coefficients are real the adjoint operators of H and G are simply their transpose (Palavajjhala *et al.*, 1994). Thus, the reconstruction of the original sequence x from sequences s and d is given as:

$$\mathrm{x} = \mathrm{H}^* \mathrm{s} + \mathrm{G}^* \mathrm{d} = \mathrm{H}^T \mathrm{s} + \mathrm{G}^T \mathrm{d} \tag{3.34}$$

3.6 Signal Decomposition using Haar Wavelet

Wavelet-based signal decomposition is carried out through MRA. The basic idea of MRA is to represent a function or signal as a limit of successive approximations.

3.6.1 *Pulse basis for infinite-length signals*

Let V_0 be the set of all square integrable signals x(t) that are constant on the unit interval $t = [i, i+1), \forall i \in Z$. It is the space of signals that can be generated from a discrete data set through zero-order hold of sampling time 1. V_0 can be expressed as

$$V_0 = \left\{ x \in L^2(R) : x(t) = \sum_{n=-\infty}^{\infty} x_n \phi(t-n), x_n \in R \right\} \tag{3.35}$$

where $\phi(t)$ is the pulse function defined as

$$\phi(t) = \begin{cases} 1 & \text{if } t \in [0,1) \\ 0 & \text{otherwise} \end{cases} \tag{3.36}$$

$\{\phi_{0,n}\}_{-\infty<n<\infty}$ with $\phi_{0,n} = \phi(t-n)$ forms an orthonormal basis set for V_0. This is commonly known as the pulse basis in digital signal processing.

Next we define V_1 as the subspace of functions in V_0 with half the original resolution. So, V_1 is the space of signals that are constant over the interval $t = [2i, 2(i+1)), \forall i \in Z$ and is defined as

$$V_1 = \left\{ x \in L^2(R) : x(t) = \frac{1}{\sqrt{2}} \sum_{n=-\infty}^{\infty} x_n \phi\left(\frac{t}{2} - n\right), \quad x_n \in R \right\} \tag{3.37}$$

$\{\phi_{1,n}\}_{-\infty<n<\infty}$ with $\phi_{1,n} = \frac{1}{\sqrt{2}} \phi(\frac{t}{2} - n)$ comprise an orthonormal basis set for V_1. This idea can be generalized to V_m, the space of signals with the resolution of $\frac{1}{2^m}$ (i.e., signals that are constant over the interval $t = [2^m i, 2^m (i+1)), \forall i \in Z$. An orthonormal basis set for V_m is $\{\phi_{m,n}\}_{-\infty<n<\infty}$ where

$$\phi_{m,n} = \frac{1}{\left(\sqrt{2}\right)^m} \phi\left(\frac{t}{2^m} - n\right) \tag{3.38}$$

3.6.2 *Pulse basis for finite-length signals*

The multiresolution concept introduced above applies to infinite domain signals. It can be extended to signals of finite length by treating them as infinite-length signals belonging to V_0 with zero values outside the finite domain (this is called as "zero-padding"). Let

$$V_0{}' = \left\{ x \in L^2(R) : x(t) = \sum_{n=0}^{2^l - 1} x_{0,n} \phi_{0,n} \right\} \tag{3.39}$$

V_0^l constitutes signals that are constant on the unit interval $t = [i, i+1)$ for $0 \le i < 2^l - 1$ and zero-padded for $-\infty < t < 0$ and $2^l \le t < \infty$. It can be thought of as the set of zero-order-hold generated signals of finite length 2^l. $\{\phi_{0,n}\}_{0 \le n < 2^l - 1}$ is the pulse basis set commonly used to represent such a signal in digital control and is shown in Figure 3.12.

Signal $x(t)$ is often represented by a 2^l-dimensional vector $[x_{0,0} \quad x_{0,1} \quad \cdots \quad x_{0,2^l-1}]^T$. The subspaces of V_0^l can be defined as follows:

$$V_1{}' = \left\{ x \in L^2(R) : x(t) = \sum_{n=0}^{2^{l-1}-1} x_{1,n} \phi_{1,n} \right\}$$

$$V_2^{l} = \left\{ x \in L^2(R) : x(t) = \sum_{n=0}^{2^{l-2}-1} x_{2,n}\phi_{2,n} \right\}$$

$$\vdots \quad \vdots \qquad\qquad\qquad \vdots$$

$$V_l^{l} = \left\{ x \in L^2(R) : x(t) = x_{l,0}\phi_{l,0} \right\} \tag{3.40}$$

Here, $\cdots \supset V_{-1}^{l} \supset V_0^{l} \supset V_1^{l} \supset \cdots \supset V_l^{l}$. In practice, we do not have signals of infinitely fine resolution. The fastest sampling rate determines the finest available resolution, i.e., resolution of V_0^l. Any signal $x \in V_0^l$ can be represented using pulse basis as:

$$x = \sum_{n=0}^{2^l-1} c_{0,n}\phi_{0,n} \tag{3.41}$$

where, $$c_{0,n} = \left\langle x, \phi_{0,n} \right\rangle = \int_0^{2^l-1} x\phi(t-n)dt$$

3.6.3 *Haar basis for finite-length signals*

Approximation of a signal of higher resolution by a signal of lower resolution is shown in Figure 3.13, where x_1^* is the lost information while approximating $x \in V_0^l$ by $x_1 \in V_1^l$. The subspace V_1^l is the coarser approximation of the space V_0^l with half its resolution and W_1^l represents the subspace containing the lost information as we go from V_0^l to V_1^l. So, V_0^l can be expressed as the "direct sum" of the orthogonal subspaces V_1^l and W_1^l:

$$V_0^l = V_1^l \oplus W_1^l$$

The basis for V_0^l is $\{\phi_{0,n}\}_{0 \le n < 2^l - 1}$ and that for V_1^l is $\{\phi_{1,n}\}_{0 \le n < 2^{l-1}-1}$. The set of functions $\{\psi_{1,n}\}_{0 \le n < 2^{l-1}-1}$ with $\psi_{1,n} = \frac{1}{\sqrt{2}}\psi\left(\frac{t}{2} - n\right)$ forms an orthonormal basis for W_1^l. Here, $\psi(t)$ is the Haar wavelet defined by Equation 3.18. So, any signal $x \in V_0^l$ can be represented in terms of $\phi_{1,n}$ and $\psi_{1,n}$ as:

$$x = \sum_{n=0}^{2^{l-1}-1} c_{1,n}\phi_{1,n} + \sum_{n=0}^{2^{l-1}-1} d_{1,n}\psi_{1,n} \tag{3.42}$$

where,

$$c_{1,n} = \langle x, \phi_{1,n} \rangle = \frac{1}{\sqrt{2}} \int_{0}^{2^{l-1}-1} x\phi\left(\frac{t}{2} - n\right)dt$$

$$d_{1,n} = \langle x, \psi_{1,n} \rangle = \frac{1}{\sqrt{2}} \int_{0}^{2^{l-1}-1} x\psi\left(\frac{t}{2} - n\right)dt$$

The orthogonality ensures that $x_1 = \sum_{n=0}^{2^{l-1}-1} c_{1,n}\phi_{1,n}$ represent the "best" approximation of x with a function half its resolution and $x_1^{*} = \sum_{n=0}^{2^{l-1}-1} d_{1,n}\psi_{1,n}$ is the "lost" information as we go from the finer to coarser resolution.

The above result can be extended to orthogonal decomposition of any V_{m-1}^{l} to $V_m^l \oplus W_m^l$. The orthonormal basis for V_m^l is $\{\phi_{m,n}\}_{0 \le n < 2^{l-m}-1}$ given by Equation 3.38 and that of W_m^l is $\{\psi_{m,n}\}_{0 \le n < 2^{l-m}-1}$ given as follows:

$$\psi_{m,n} = \frac{1}{(\sqrt{2})^m} \psi\left(\frac{t}{2^m} - n\right) \tag{3.43}$$

Applying this idea successively from V_0^l, we can write

$$\begin{aligned} V_0^l &= V_1^l \oplus W_1^l \\ &= V_2^l \oplus W_2^l \oplus W_1^l \\ &\vdots \quad \vdots \\ &= V_l^l \oplus W_l^l \oplus W_{l-1}^l \oplus \ \cdots \ W_1^l \end{aligned}$$

So, the 2^l orthonormal basis functions for V_0^l can be chosen as $\left\{\phi_{l,0}, \{\psi_{m,n}\}_{\substack{m=1,\cdots,l \\ n=0,\cdots,2^{l-m}-1}}\right\}$.

The Haar basis for V_0^3 is $\{\phi_{3,0}, \psi_{3,0}, \{\psi_{2,n}\}_{n=0,1}, \{\psi_{1,n}\}_{n=0,1,2,3}\}$ and is shown in Figure 3.14.

Any signal $x \in V_0^l$ can be represented using Haar basis as:

$$x = c_{l,0}\phi_{l,0} + \sum_{m=1}^{l} \sum_{n=0}^{2^{l-m}-1} d_{m,n}\psi_{m,n} \tag{3.44}$$

The Haar decomposition for the finite-length signal can be viewed as a change of basis from the pulse basis (Equation 3.41) to the wavelet basis (Equation 3.44). This change of basis is a unitary transformation as both are complete orthonormal basis sets for V_0^l (Lee *et al.*, 1992). The two representations (Equations 3.41 and 3.44) are related through a unitary transformation $\tilde{x}^* = U\,\tilde{x}$ where

$$\tilde{x}^* = [c_{l,0} \quad d_{l,0} \quad d_{l-1,0} \quad d_{l-1,1} \quad \cdots \quad \cdots \quad d_{1,2^{l-1}-2} \quad d_{1,2^{l-1}-1}]^T$$

$$\tilde{x} = [c_{0,0} \quad c_{0,1} \quad c_{0,2} \quad c_{0,3} \quad \cdots \quad \cdots \quad c_{0,2^l-2} \quad c_{0,2^l-1}]^T$$

and U is a 2^l by 2^l orthonormal coefficient matrix, obtained from the scaling function and dilation and translation of the wavelet. For $x \in V_0^3$, we have the following unitary transformation:

$$\begin{bmatrix} c_{3,0} \\ d_{3,0} \\ d_{2,0} \\ d_{2,1} \\ d_{1,0} \\ d_{1,1} \\ d_{1,2} \\ d_{1,3} \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} \\ \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{-1}{\sqrt{8}} & \frac{-1}{\sqrt{8}} & \frac{-1}{\sqrt{8}} & \frac{-1}{\sqrt{8}} \\ \frac{1}{2} & \frac{1}{2} & \frac{-1}{2} & \frac{-1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & \frac{-1}{2} & \frac{-1}{2} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} c_{0,0} \\ c_{0,1} \\ c_{0,2} \\ c_{0,3} \\ c_{0,4} \\ c_{0,5} \\ c_{0,6} \\ c_{0,7} \end{bmatrix}$$

The first row of U corresponds to the scaling function $\phi_{0,l}$ and the other rows correspond to sampled version of Haar wavelet at different dilations and translations.

3.7 Conclusions on Introduction to Wavelets

The most appealing properties of wavelets are their time-frequency localization and multiresolution approximation capabilities. Projections of functions or signals on the wavelet bases result in decompositions, which provide time-frequency information together. These decompositions along with the attractive computational properties of wavelets are exploited in process control applications.

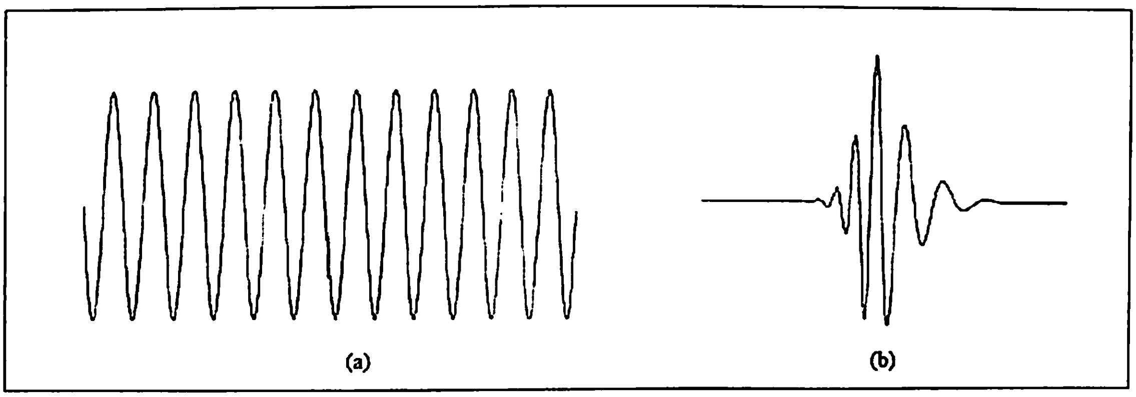


Figure 3.1 Demonstration of (a) a Wave and (b) a Wavelet. (Burrus *et al.*, 1998)

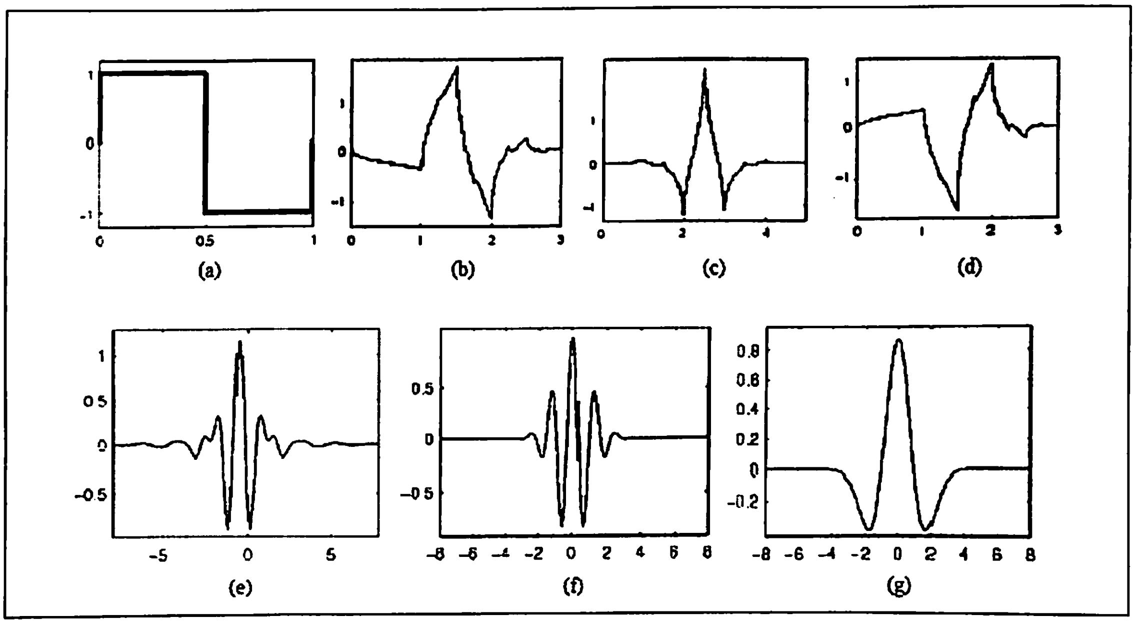


Figure 3.2 Wavelet families (a) Haar (b) Daubechies 4 (c) Coiflet 1 (d) Symlet 2 (e) Meyer (f) Morlet (g) Mexican Hat. (Sripathi, 2003)

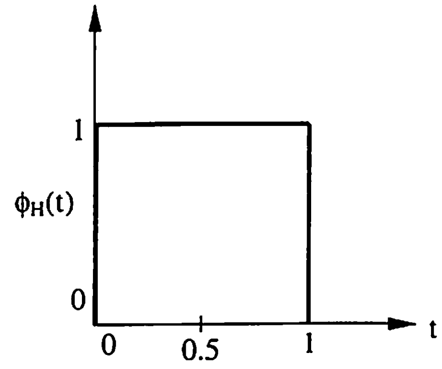


Figure 3.3 Haar Scaling function.

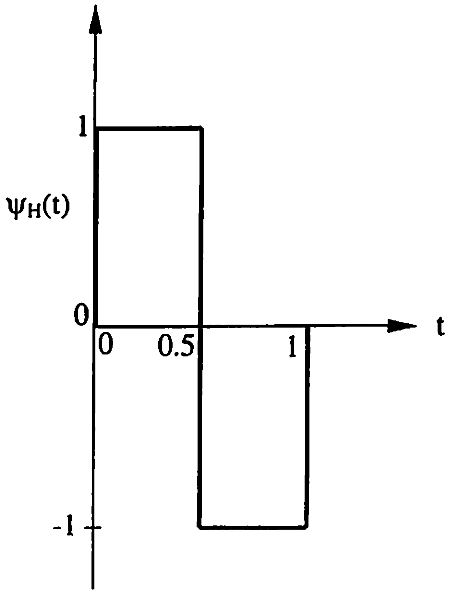


Figure 3.4 Haar Wavelet.

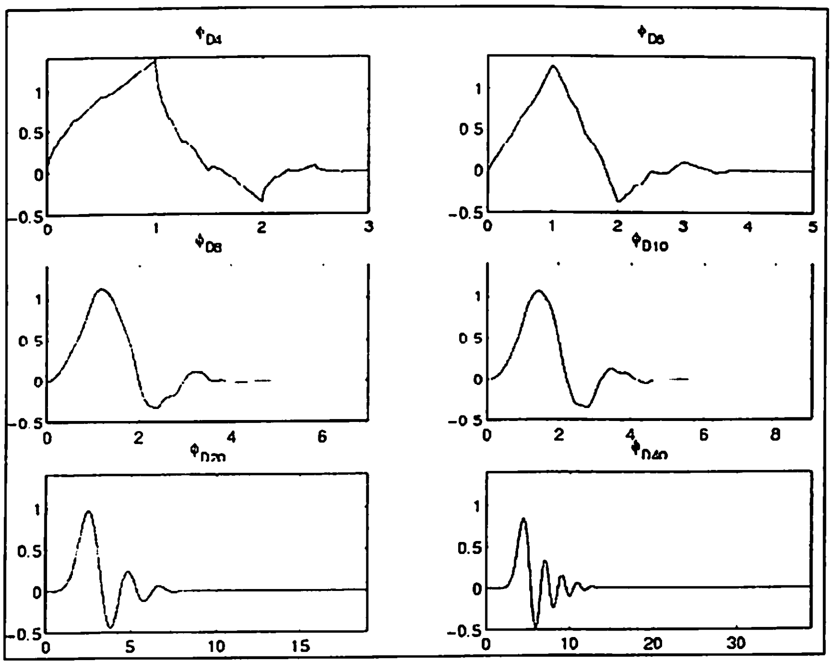


Figure 3.5 Daubechies Scaling functions. (Qiao and Milam, 2003-2006)

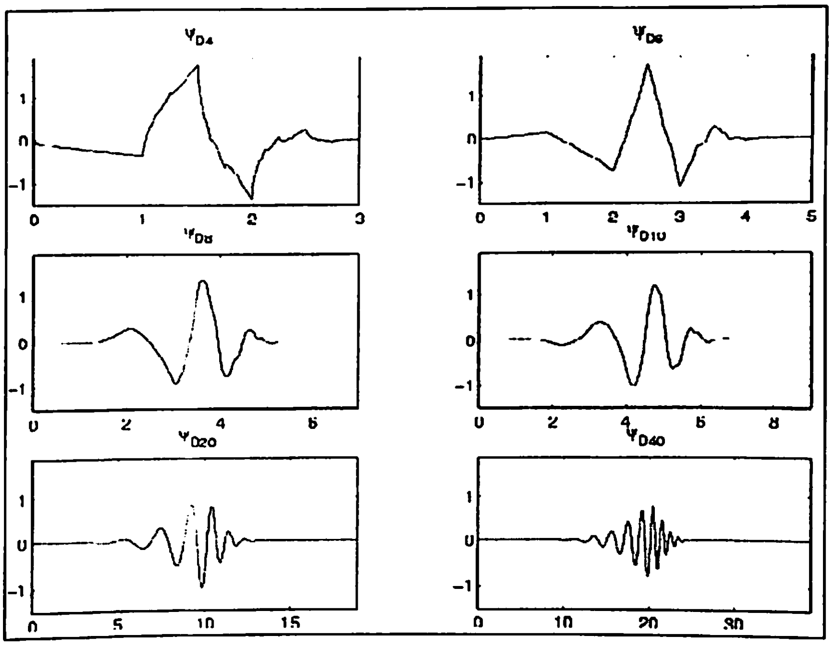


Figure 3.6 Daubechies Wavelet functions. (Qiao and Milam, 2003-2006)

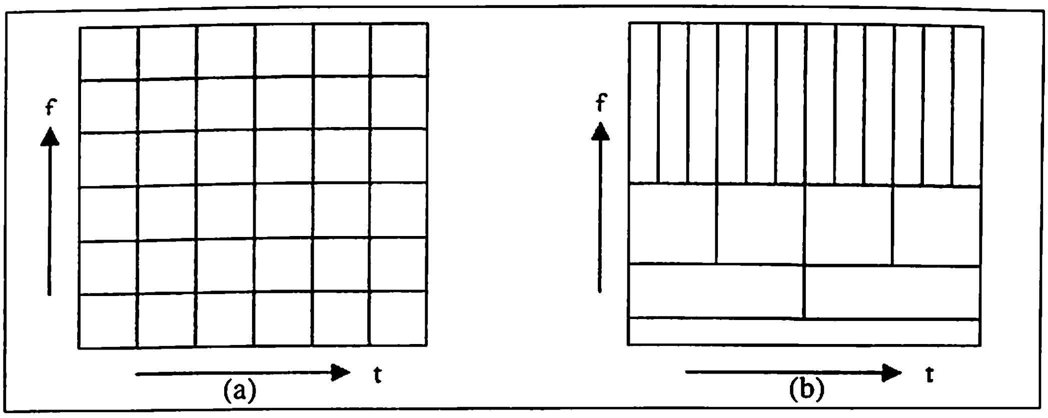
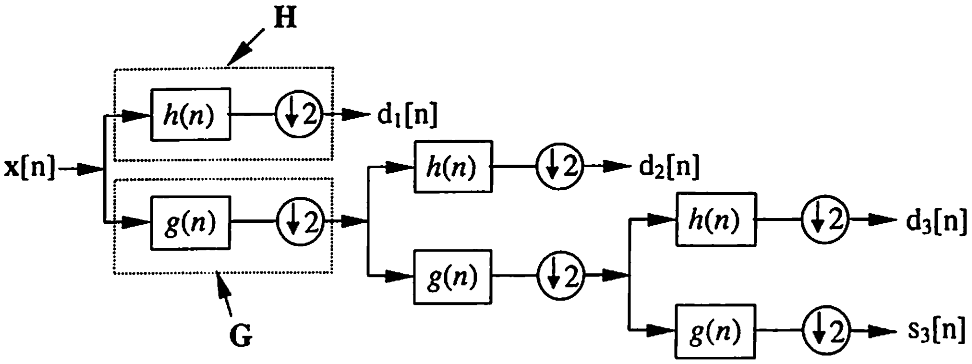
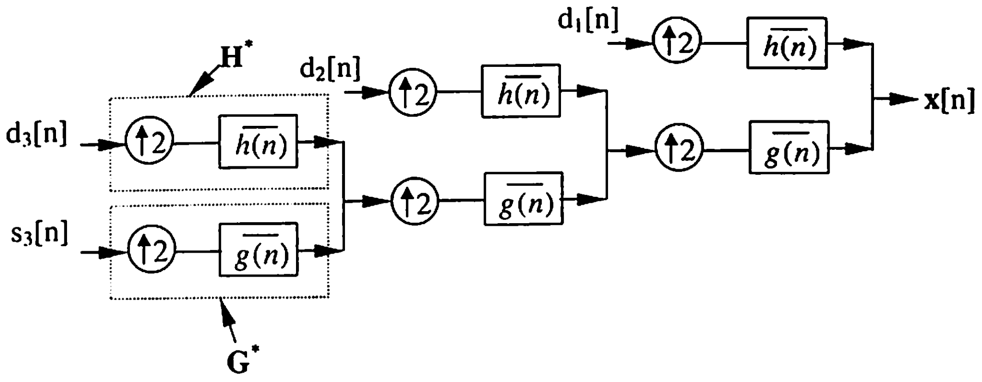


Figure 3.7 The Time-Frequency tiling for (a) STFT and (b) WT. (Sripathi, 2003)



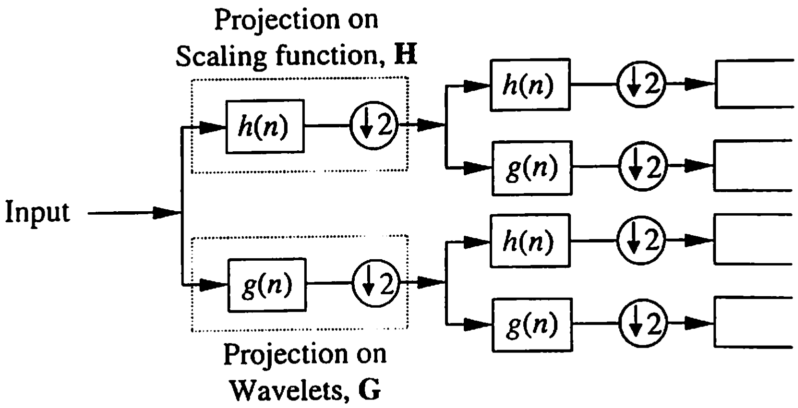
(↓2 means downsample by 2 (drop even terms))

Figure 3.8 Three-level discrete wavelet decomposition of a signal x.



($\overline{h(n)}$ is the complex conjugate of $h(n)$,↑2 means upsample by 2 (pad even terms with 0))

Figure 3.9 Three-level discrete wavelet reconstruction of a signal **x**.



(↓2 means downsample by 2 (drop even terms))

Figure 3.10 Wavelet Packet Decomposition.

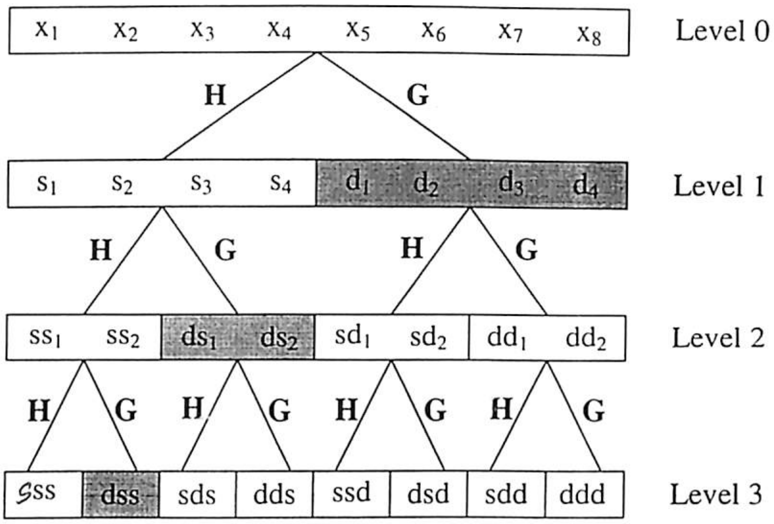


Figure 3.11 Wavelet packet decomposition of a sequence. (Wickerhauser, 1991)

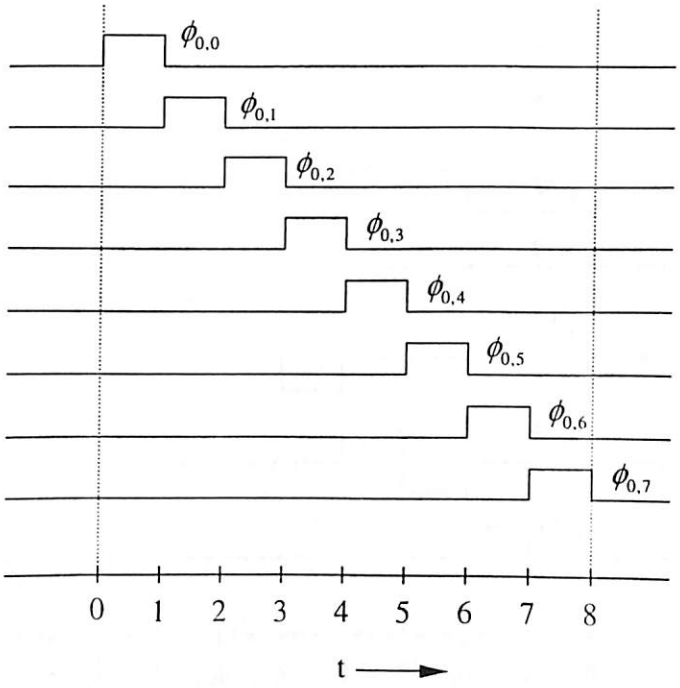
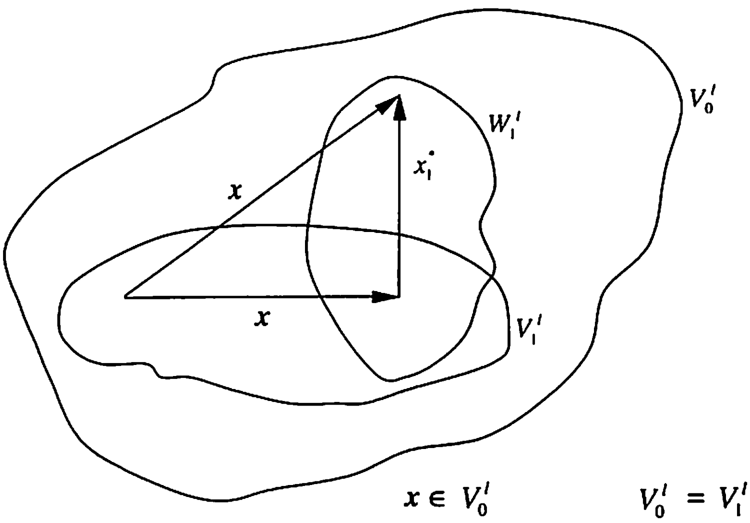


Figure 3.12 Pulse basis for finite-length signals.



$x \in V_0'$ $\qquad V_0' = V_1' \oplus W_1'$

$x_1 \in V_1'$ $\qquad V_1' \subset V_0'$ & $V_1' \perp W_1'$

$x_1^* \in W_1'$ $\qquad x = x_1 + x_1^*$

Figure 3.13 Approximation of signal **x** in vector space V_1.

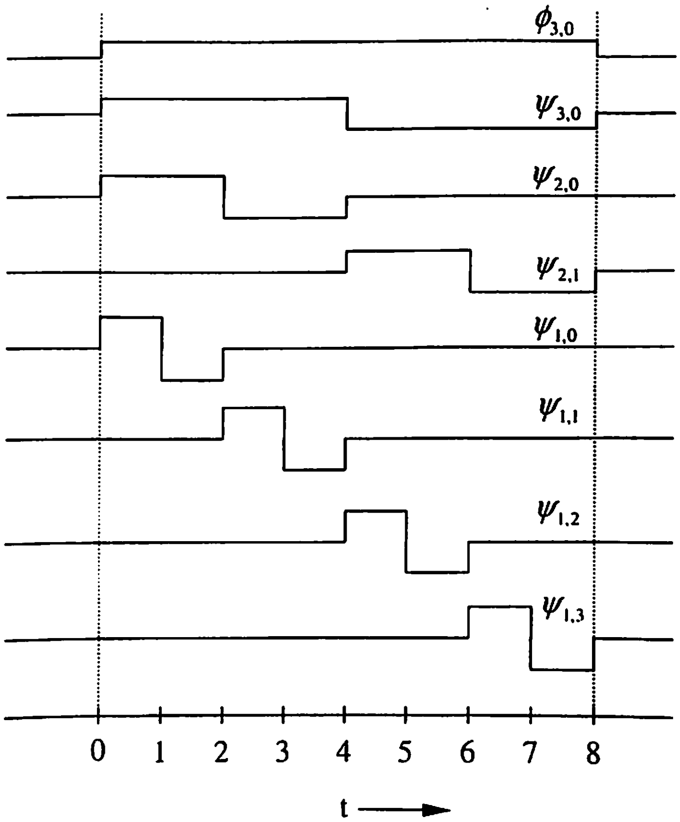


Figure 3.14 Haar basis for finite-length signals.

CHAPTER 4

EXPERIMENTAL SETUP

This chapter provides the description of the experimental setups in brief along with the schematic diagrams. The hardware used is explained with relevant figures in addition with the setups. The photographs of the complete experimental setups along with accessories are also given.

4.1 Heat Exchanger Unit and Accessories

A plate type non-mixing heat exchange unit (Model PCT 13), supplied by Feedback Instruments Ltd. (UK) is chosen for experimentation (Figure 4.1). The line diagram of the unit is shown in Figure 4.2. The process fluid (cold water) is supplied from an overhead tank and is fed to the unit through flexible tubing, a manual flow control valve and a rotameter. The range of rotameter is 0 to 280 cc/min. The process fluid is then passed through a non-mixing, plate type heat exchanger, where it is being heated by a hot water stream that flows counter currently. The outlet process fluid is discharged to a suitable sink. The inlet and outlet temperatures of the process fluid are measured by separate thermocouples. The hot water (heating medium) is re-circulated from a tank housed within the setup, filled with a heater and thermostat. The temperature of the water in the tank is set at 80°C using the thermostat. Hot water is pumped to the heat exchanger through a manual control valve; a rotameter and a stepper motor actuated control valve. The manual control valve and the rotameter in the hot water line are similar to that of a process fluid line. The motorized valve is diaphragm type valve operated by a 12V DC stepper motor with 1.8 degree rotation per pulse. Its action is highly repetitive. Two thermocouples are provided to measure temperature of heating fluid, one in the tank and the other at the outlet of the heat exchanger. The four chromel-alumel thermocouples used are identical and are provided with compensating cables. The manipulated variable is hot water flow-rate and final control element is the motorized valve. Both regulatory and servo problems are studied. The regulatory problem is to maintain the temperature of the outlet process stream at its desired value against the variations in the inlet temperature and flow rate. The servo performance has been investigated with set point step changes in the temperature of the process fluid. The temperature of the inlet process fluid changes with ambient temperature, generally within range 20 to 35°C. During experiments, the flow rate of the process fluid is kept within range 100 to 280 cc/min.

A specially designed electrical console (Model PCT 10), supplied by the same manufacturer is used along with the heat exchanger unit (Figure 4.3). It is basically a semi-portable modular type control panel. The electrical console houses several independently operable instruments and are also equipped with certain safety features as described below:

- **Voltmeter**: One voltmeter with digital display having range 0-1 V DC and least count 0.001V.
- **Ammeter**: One ammeter with analog display having range 4-20mA and least count 1mA.
- **Signal conditioner**: Two independent signal conditioners with plug-in attachments for specific applications. The output signal can be obtained both as a current signal [4-20mA] or voltage signal [0-1V]. In absence of any plug-in attachment the setup can be utilized to convert current signal ~~top~~ into voltage signal or vise versa, within the ranges mentioned above.
- **Relay Operator**: One relay operator with adjustable configuration of 'normally off' and 'normally on' position.
- **Power Supply Plug**: Four 240V AC and four 24V AC power supply sockets are mounted on the side panel of the electric console. These are used for power supply to the pump & heater in the experimental setup. For safety, the supply is through an automated earth leakage circuit breaker to reduce severity of an electric shock in the event of any accident.
- **Metered Output**: One analog current source, manually adjustable within range 4-20 mA.
- **Motor Positioner**: One motor positioner with stepper motor driver circuitry and amplifier. The input signal to the motor positioner is from the 'controller' or the 'metered output' as current signal in the range 4-20mA. The motor positioner communicates with motor through a 5 pin socket.
- **Process Controller**: The process controller block consists of a tunable 452 plus microprocessor and a dual display unit. Input signal to the process controller is

current signal in the range 4-20mA. The controlled output signal can be obtained as 4-20 mA DC as well as 24V AC and 240 V AC signal. The microprocessor can be interfaced with computer through COM port. The display unit has two parts, the upper part shows measured variable and controller settings and lower part shows set point. Touch sensitive control key are provided for manually changing the set point, alarm, P, I, D parameter settings, sampling rate, hysteresis level, span and zero settings etc. Most of these parameters can also be read and modified through a computer interfaced with the controller.

- **Circuit Breaker**: Two manual single-touch circuit breakers are provided to break the circuits for the 24V AC and 240V Ac power supply outputs respectively.

The color code for all DC signal flow lines are universally indicated by red for positive and black for negative sockets and connectors. The photograph of the experimental setup is shown in Figure 4.4. The experiments are conducted through direct digital control (DDC) and the microprocessor-based controller is bypassed. In this case the experimental setup is interfaced directly to the PC bus through a DT 2811-PGH AD/DA card and delivers the corresponding 0-1 V signal. The DT 2811-PGH is a typical AD/DA card for industrial process control in low noise environment and is compatible with IBM PC/XT/AT. The AD/DA card has several adjustable features and settings used are given in Table 4.1.

The computer programs used for process control are developed and are presented in Appendix I. The experiments are conducted at a sampling time of 1 sec. While each sampled data represent a statistical average of 2000 consecutive readings. Thus the utilized throughput capacity of AD/DA card is 2 kHz.

The computing machine used for direct digital control is an Intel Celeron processor with CPU 266MHz of M.C. Modi & Co. make having base memory 640K and extended memory 64512K. The operating system is DOS Version 5 and the process control codes are written in language C. The compiler used is Turbo C++ Version 3.0 made by Borland International, Inc. © 1992.

4.2 Liquid Level Unit and Accessories

In addition with the heat exchanger, also a laboratory model level unit (Model PCT 9), supplied by Feedback Instruments Ltd. (U.K.) is chosen for experimentations (Figure

4.5). The line diagram is shown in Figure 4.6. The process fluid is water, stored in a sump tank of 18 litres capacity. Water from the sump is pumped to a process tank made of Perspex, placed at a higher level (Figure 4.5 and Figure 4.6). The supply line to the process tank has a rotameter and a stepper motor operated control valve. The rotameter has a range of 0.2 to 3.0 lit/min. The process tank has two compartments separated from each other by a partition wall. There is a large circular hole in the partition wall, which can be sealed off with a stopper plug. The level of water in the right hand compartment of the tank is measured with a float operated variable resistance type level sensor. This compartment is taken as the process tank for the present work. The process tank has two drainages of different diameters fitted with solenoid valves. There is also a manually operated drainage valve apart from an over flow drainage. The area of this tank is 215cm^2 the height up to the overflow drainage is 140 mm. The measurable height of the liquid level using the level sensor in the tank is from 30 mm to 130 mm. The level 30 mm corresponds to 0 volt voltage signal (4mA current signal) and the level 130mm corresponds to 1 volt voltage signal (20mA current signal) measured using the electrical console.

The photograph of the experimental setup is shown in Figure 4.7. Signal from the level sensor is fed to the digital computer through signal conditioner. An electric console, AD/DA cards, adapter and computing machine are used in the level control experiments. The output from the computer is used to operate the stepper motor controlled valve, which is installed in the inlet supply line to the process tank. The manipulated variable is inlet flow rate of water in the process tank, while the final control element is the stepper motor control valve. Both servo and regulator problems are studied. In regulator problem, the level of water in the process tank is maintained at its operating value in presence of disturbance. The disturbances are introduced by opening the drainage pipelines. In servo problem the aim is to change the level of the water in the process tank to a new set point value. In the present work a step change of 30 mm (from 80 mm to 110 mm) in the set point is considered.

Table 4.1 Characteristics of DT2811 Card.

DT2811-PGH Mode	**Operation**	**Range**	**Resolution in bit**	**Gain**	Maximum throughput
Differential Input mode	AD	±5 V	12	1	20 kHz
	DA	±5 V	12	1	50 kHz

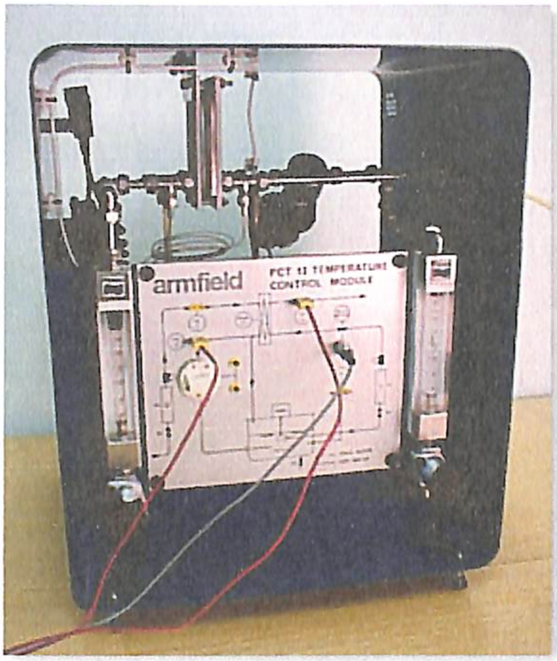


Figure 4.1 Photograph of the Heat Exchanger Unit.

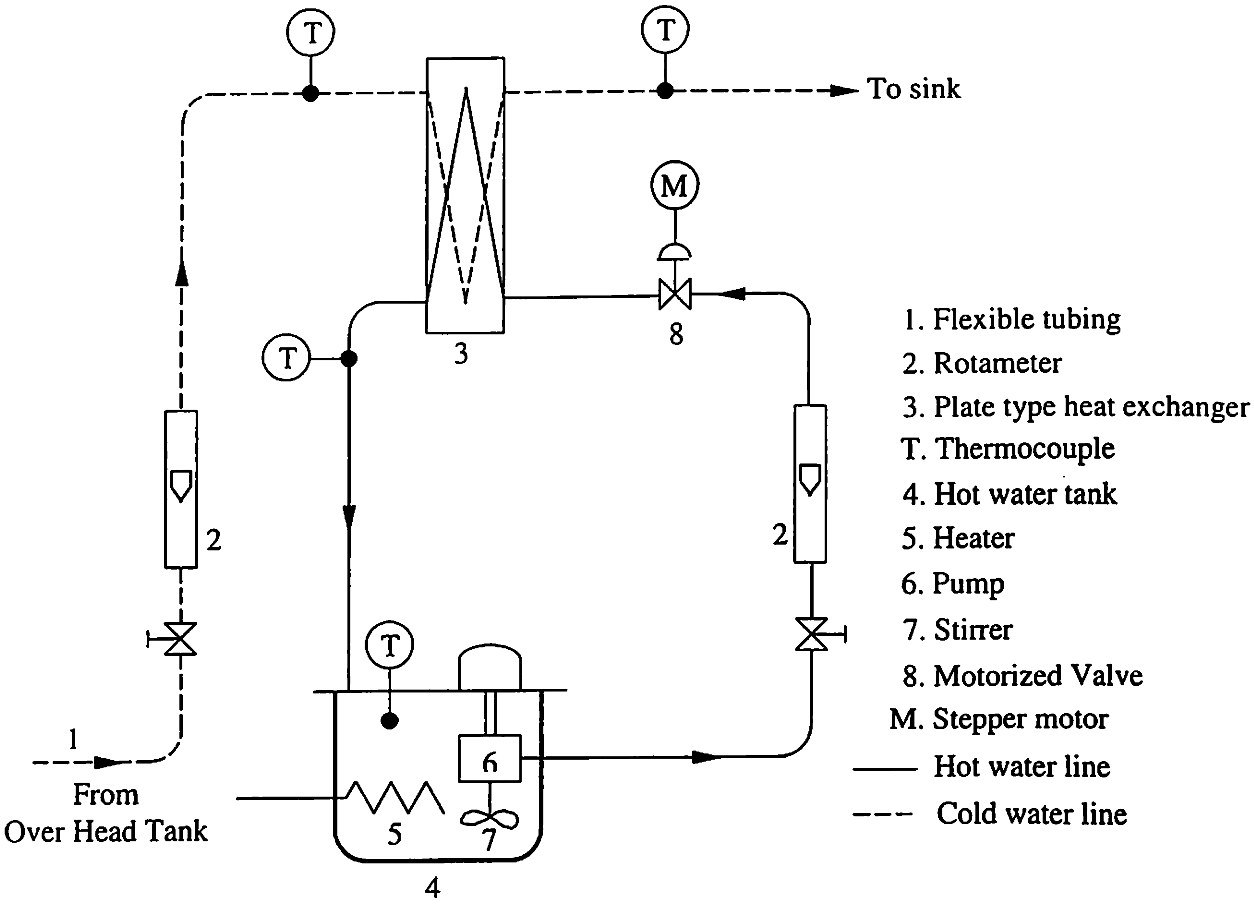


Figure 4.2 Line diagram of the Heat Exchanger Unit.

Figure 4.3 Photograph of the Electrical Console.

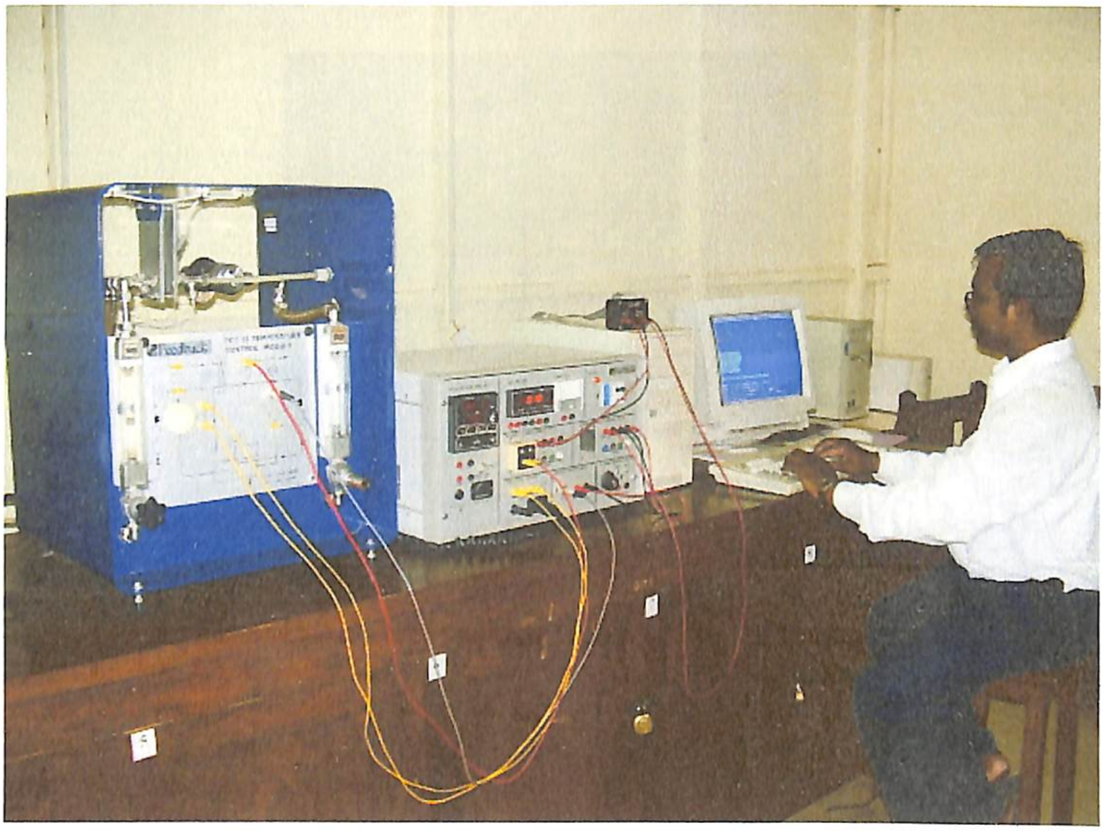


Figure 4.4 Photograph of the experimental setup for temperature control in the Heat Exchanger Unit.

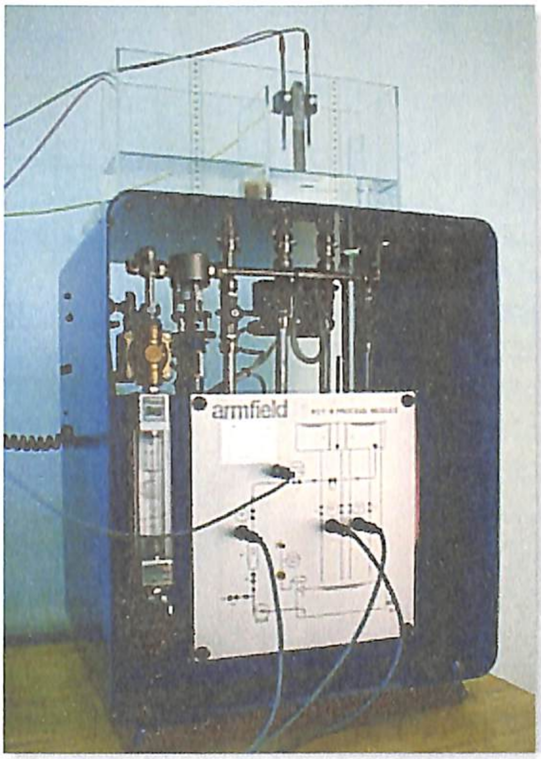
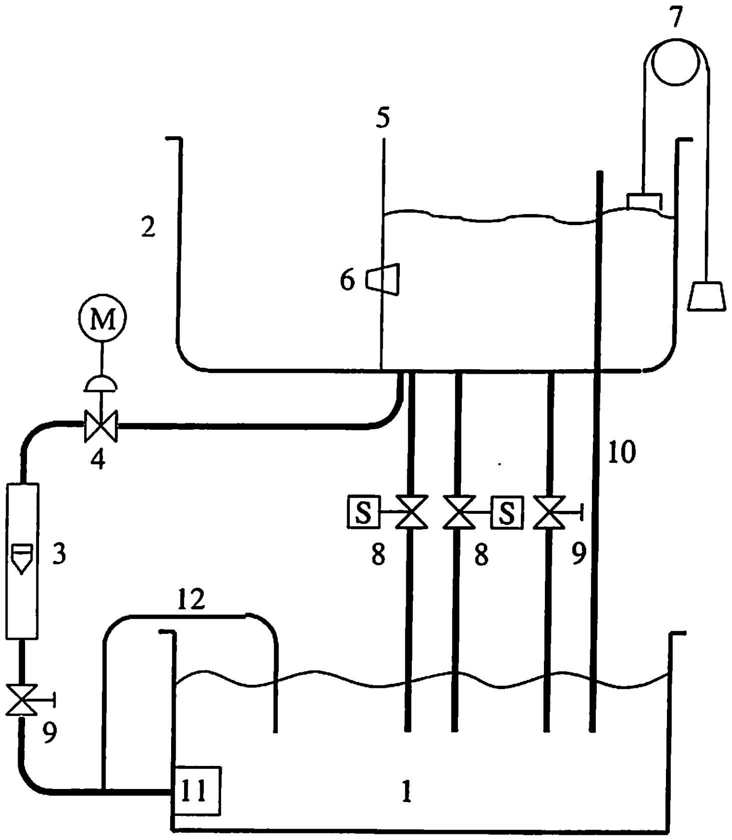


Figure 4.5 Photograph of the Liquid Level Unit.



1. Sump tank
2. Process tank
3. Rotameter
4. Motorized valve
M. Stepper motor
5. Process tank partition wall
6. Stopper plug
7. Float operated level sensor
8. Solenoid operated valve
9. Manual valve
10. Overflow drainage
11. Pump
12. Bypass line (flexible tubing)

Figure 4.6 Line diagram for the Liquid Level Unit.

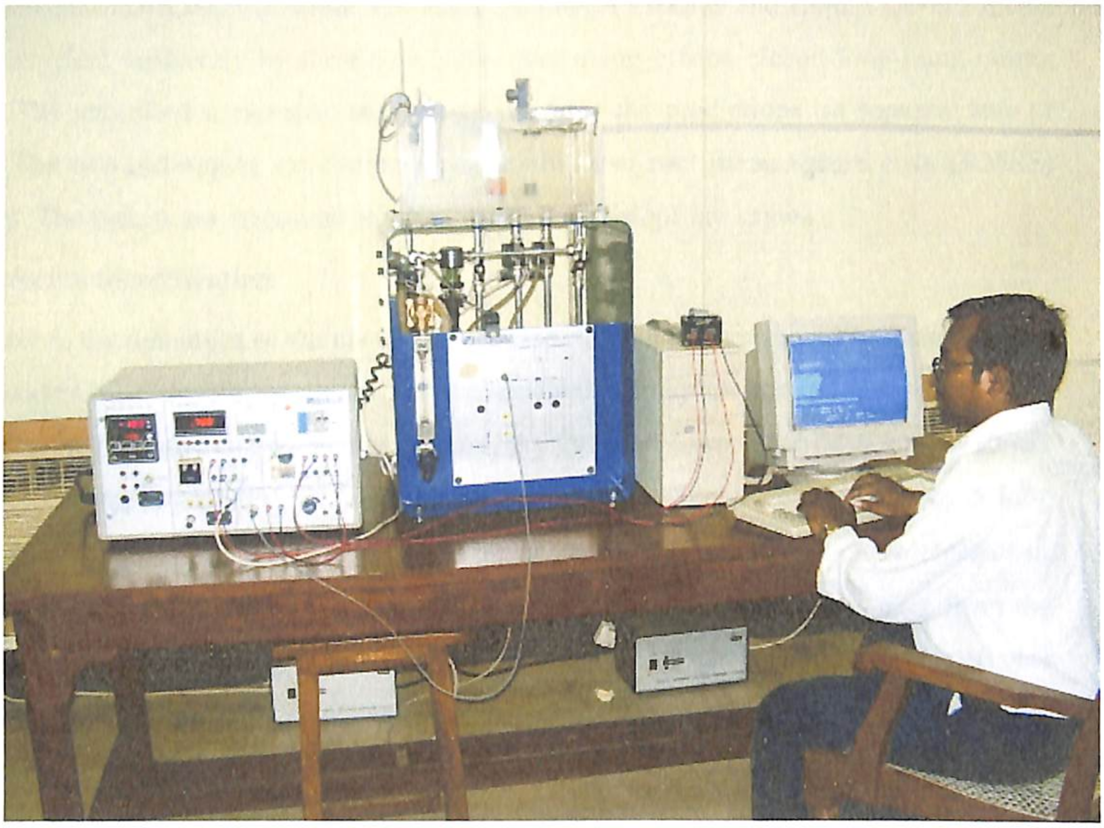


Figure 4.7 Photograph of the experimental setup for level control in the Liquid Level Unit.

CHAPTER 5

WAVELET-BASED IDENTIFICATION OF CHEMICAL PROCESSES

This chapter discusses the need for process identification and the theory behind classical and wavelet-based identification. The Heat Exchanger Process and Liquid Level Process are identified separately by these two techniques using offline closed-loop input-output data. The identified parametric models are used for the predictions on separate sets of data. The two techniques are compared on the basis of root mean square error (RMSE) values. The results are discussed and finally the conclusions are drawn.

5.1 Process Identification

In general, the dynamics of the chemical processes are not known precisely. In such cases the models that developed based on fundamental principles are quite inadequate to describe the dynamic characteristics of the real processes. Even if we have a good steady state model for a process, it may not be sufficient for effective control during a long operation, because the chemical processes are in general nonlinear (i.e. characteristics of the corresponding linearized models change when the operating point changes from the point of linearization) and some are nonstationary (i.e. their dynamic characteristics change with time because of change in the value of their important physical or chemical parameters with time) in nature (Stephanopoulos, 1993). Sometimes the theoretical models of the chemical processes are too complicated to be useful, especially for controller design proposes.

In all these cases, whether the theoretical modeling is impossible, impractical or inconvenient, the useful model of the system can be obtained by process identification. Process identification involves constructing a process model strictly from experimentally obtained input-output data, with no recourse to any laws concerning the fundamental nature and properties of the system (Ogunnaike and Ray, 1994).

5.2 Classical Least-Square Identification (CLSI)

Let a poorly known process be described by the following linear difference equation of order *k*:

$$\hat{x}_n = a_1 x_{n-1} + a_2 x_{n-2} + \cdots + a_k x_{n-k} + b_1 m_{n-1} + b_2 m_{n-2} + \cdots + b_k m_{n-k} \tag{5.1}$$

where $\hat{x}_n$ is the *predicted* value of the output at nth instant; x_i and m_i are process output and input values respectively *measured* at i^{th} sampling instant and $a_1, a_2, \cdots, a_k, b_1, b_2, \cdots, b_k$ are constants but imprecisely known parameters (to be identified).

At nth instant error between the measured value of the output, x_n and the predicted value, $\hat{x}_n$ is given by.

$$\varepsilon_n = x_n - \hat{x}_n = x_n - (a_1 x_{n-1} + a_2 x_{n-2} + \cdots + a_k x_{n-k} + b_1 m_{n-1} + b_2 m_{n-2} + \cdots + b_k m_{n-k}) \tag{5.2}$$

Our objective is to minimize the sum of the squares of the errors at all (say N) sampling instants to obtain the best values of the poorly known parameters. Thus the estimate of the process parameters is given by the solution of the following least-square problem:

$$\min J = \sum_{n=1}^{N} \{x_n - a_1 x_{n-1} - a_2 x_{n-2} - \cdots - a_k x_{n-k} - b_1 m_{n-1} - b_2 m_{n-2} - \cdots - b_k m_{n-k}\}^2 \tag{5.3}$$

The necessary conditions, which must be satisfied at the point where J is a minimum:

$$\frac{\partial J}{\partial a_1} = \frac{\partial J}{\partial a_2} = \cdots = \frac{\partial J}{\partial a_k} = \frac{\partial J}{\partial b_1} = \frac{\partial J}{\partial b_2} = \cdots = \frac{\partial J}{\partial b_k} = 0 \tag{5.4}$$

Equation 5.4 gives a set of algebraic equations. The solutions to these equations can be written in the following matrix form (Edgar and Himmelblau, 1989; Gupta, 1995; Luyben, 1990; Ljung, 1996):

$$\underline{\theta} = [\underline{\underline{A}}^T \underline{\underline{A}}]^{-1} \underline{\underline{A}}^T \underline{x} \tag{5.5}$$

where,

$$\underline{\underline{A}} = \begin{bmatrix} x_{k-1} & x_{k-2} & \cdots & x_0 & m_{k-1} & m_{k-2} & \cdots & m_0 \\ x_k & x_{k-1} & \cdots & x_1 & m_k & m_{k-1} & \cdots & m_1 \\ x_{k+1} & x_k & \cdots & x_2 & m_{k+1} & m_k & \cdots & m_2 \\ x_{k+2} & x_{k+1} & \cdots & x_3 & m_{k+2} & m_{k+1} & \cdots & m_3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{N-1} & x_{N-2} & \cdots & x_{N-k} & m_{N-1} & m_{N-2} & \cdots & m_{N-k} \end{bmatrix} \tag{5.6}$$

$$\underline{x} = [x_k \quad x_{k+1} \quad x_{k+2} \quad \cdots \quad x_N]^T \tag{5.7}$$

$$\underline{\theta} = [a_1 \quad a_2 \quad \cdots \quad a_k \quad b_1 \quad b_2 \quad \cdots \quad b_k]^T \tag{5.8}$$

Here, $\underline{\underline{A}}$ is the data matrix, which contains measured values of the output and input at different instants, $\underline{x}$ is the vector of the measured outputs and $\underline{\theta}$ is the vector of the poorly known parameters which are identified using equation (5.5).

Generally for process identification, a known input change (e.g. step, pulse, sine) is introduced to the process and its output is recorded. Sometimes a pseudo-random binary sequence of inputs (the input changes randomly between two values) is used (Bequette, 2003).

For identifying the parameters of the system, the model order, k is to be known beforehand. As a general starting point one could employ first- or second-order models with or without dead time. The complex models of high order may not necessarily produce better controller design and will burden the computational effort without tangible results. There exist a surprisingly large number of processes, which could be effectively described by such low-order models (Stephanopoulos, 1993).

5.2.1 *Algorithm for Classical Least-Square Identification*

The following computational steps are involved in classical least-square identification (CLSI).

1. Assume a first-order (k = 1) model for the system
2. Formulate the data matrix $\underline{\underline{A}}$ using Equation 5.6
3. Formulate the measured output vector $\underline{x}$ using Equation 5.7
4. Obtain the vector of the parameters $\underline{\theta}$ using Equation 5.5
5. Find out the sum of the square errors, J using Equation 5.3
6. Increase the order k by 1 and go to step 2 through step 5
7. Compare the values of J with order k and that with order k-1. If the recent value (with order k) is more than the previous one (with order k-1), take the model order as k-1 and values of the parameters identified with this order. Otherwise go to step 6.

A source code CLSI.C (Appendix I) is developed for classical least-square identification of the chemical processes from the off-line input-output data.

5.3 Wavelet-based Least-Square Identification (WLSI)

The raw data collected from the process plant usually consists of the cumulative effects of noise, process disturbances, instrument inaccuracies and uncertainties, process drifts and process events. The empirical models developed from such noisy data by off-line identification can give inaccurate prediction and misleading conclusion. Palavajjhala *et al.* (1996) suggest that wavelet can be used as a prefilter to separate the useful information from noise before the parameters are identified. This approach is based on system identification algorithms that belong to the method of modulating functions (Shinbrot, 1957). In the present work, Daubechies wavelet filter with four coefficients is used for process identification.

5.3.1 *Daubechies Wavelet Filter Coefficients*

A particular set of wavelets is specified by a particular set of numbers, called *wavelet filter coefficients*. Daubechies wavelet filter coefficients is a class which includes members ranging from highly localized to highly smooth. The simplest and most localized member, often called DAUB4, has only four coefficients, $c_0,\ldots,c_3$. The values of these coefficients ~~are~~ were first found out by Daubechies (Press *et al.*, 1992) and are given below.

$$c_0 = \frac{1+\sqrt{3}}{4\sqrt{2}}, \qquad c_1 = \frac{3+\sqrt{3}}{4\sqrt{2}}, \qquad c_2 = \frac{3-\sqrt{3}}{4\sqrt{2}}, \qquad c_3 = \frac{1-\sqrt{3}}{4\sqrt{2}} \tag{5.9}$$

A wavelet coefficient matrix, $\underline{\underline{W}}$ is constructed from these coefficients as follows:

$$\underline{\underline{W}} = \begin{bmatrix} c_0 & c_1 & c_2 & c_3 & & & & & & \\ c_3 & -c_2 & c_1 & -c_0 & & & & & & \\ & & c_0 & c_1 & c_2 & c_3 & & & & \\ & & c_3 & -c_2 & c_1 & -c_0 & & & & \\ \vdots & \vdots & & & & & \ddots & & & \\ & & & & & & c_0 & c_1 & c_2 & c_3 \\ & & & & & & c_3 & -c_2 & c_1 & -c_0 \\ c_2 & c_3 & & & & & & & c_0 & c_1 \\ c_1 & -c_0 & & & & & & & c_3 & -c_2 \end{bmatrix} \tag{5.10}$$

The blank entries in the right hand side of equation (5.10) signify zeros. The matrix $\underline{\underline{W}}$ is pre-multiplied to a column vector of data points for obtaining the discrete wavelet transform of the data set. The first row of the matrix generates one component of the data convolved with the filter coefficients $c_0,\ldots,c_3$. Similarly the third, fifth, and other odd rows. The even rows perform a different convolution, with coefficients $c_3,-c_2,c_1,-c_0$. The action of the matrix is to perform two related convolutions, then to decimate each of them by half (through away half the values), and interleave the remaining halves. The filter $c_0,\ldots,c_3$ is used to smoothen the data vector and is thus called as a smoothing or low-pass filter, ***H***. The filter $c_3,-c_2,c_1,-c_0$ is used to collect the data's lost or detail information (high frequency components) and thus called as a high-pass filter, ***G***. The smooth components are denoted as s-components and the detail components are denoted as **d**-components.

5.3.2 *Discrete Wavelet Transform*

The discrete wavelet transform (DWT) consists of applying a wavelet coefficient matrix hierarchically, first to the full data vector of length *N*, then to the "smooth" vector of length *N* / 2, then to the "smooth-smooth" vector of length *N* / 4, and so on until only a trivial number of "smooth-...-smooth" components (usually 2) remain. This procedure is called a *pyramidal algorithm*. The output of the DWT consists of these remaining components and all the "detail" components that were accumulated along the way. Figure 5.1 shows the DWT of a discrete signal consisting of 16 data points. In this figure *s*'s are the smooth components, *d*'s are the detailed components, *ss*'s are the smooth-smooth components, *ds*'s are detail-smooth components, *sss*'s are smooth-smooth-smooth components and *dss*'s are detail-smooth-smooth components of the original signal (**x**). It can be noted that once the *d*'s are generated, they simply propagate through to all subsequent stages.

5.3.3 *Algorithm for Wavelet-based Least-Square Identification*

The following algorithm is used for Wavelet-based Least-Square Identification (WLSI):

1. Obtain the input-output data from an off-line identification experiment.

2. Divide the whole data set into two parts. Use first NT data points for parameter identification and the second NP data points for model validation.
3. At scale m=0 (which represents raw noisy data), identify the model parameters with NT data points from the classical least squares identification technique and predict up to NP number of points in the future horizon.
4. At scale m=1, apply wavelet algorithm and filter up to first level of scaling. Identify the model with this data and predict in future horizon.
5. Calculate RMSE index as:

$$RMSE = \sqrt{\sum_{i=1}^{NP} \frac{(\hat{x}_i - x_i)^2}{NP}} \tag{5.11}$$

   where, $\hat{x}$ and x are predicted and actual values respectively. NP is the number of data points in the prediction horizon.
6. Compare the RMSE value with the previous case. Stop when the RMSE value exceeds that in the previous case and take the previous m as the optimum scale. Otherwise go to step 7.
7. Increase the value of m by 1 and repeat the same procedure as given in step 4 through step 6.

A source code WLSI.C (Appendix I) is developed for wavelet-based identification of the chemical processes from the off-line input-output data.

5.4 Results and Discussions

5.4.1 *Identification of the Heat Exchanger Process*

The heat exchanger process is described in section 4.1. This is a single-input-single-output (SISO) system having input as the hot water flow rate (in cc/min) and output as the cold water outlet temperature (in °C). The cold water flow rate is kept constant at 200 cc/min. The cold water inlet temperature is assumed to be constant at the room temperature. The experimental setup has a water tank where distilled water is heated and its temperature is held at 80±2°C using thermostat and the heating coil. The hot water is pumped from the tank and passes through the manual control valve, rotameter, motorized valve, the plate-type heat exchanger and finally to the hot water tank for re-circulation. The cold water comes from the overhead tank and its flow rate is regulated using manual control valve and the rotameter. After passing through the plate-type heat exchanger where it gains heat from hot water it is rejected to the sink (Figure 4.2). The process is brought to steady state with the hot water and cold water flow rates of 50 ml/min and 200 ml/min respectively and outlet temperature of cold water around 35°C. For a step change of 10°C in the setpoint (cold water temperature) using an un-tuned dynamic matrix controller approximately sinusoidal inputs and outputs are generated from the experiment. The details of applying dynamic matrix controller are discussed in Chapter 6. These input-output data are used for identification of the heat exchanger process. A typical input-output raw data is shown in Figure 5.2. The denoised data using wavelet filter is also shown along with these data.

There are 300 values in the raw noisy input-output data. First 200 values are used for parameter identification and the last 100 values are used to compare the predicted results by classical least-square and wavelet-based methods. Initially a first order model is assumed. This model gives a higher RMSE compared to the second order model (Figure 5.3). A third order model could not be tested due to computational limitations. The model identified using classical least square method is

$$\hat{x}_n = 0.7129x_{n-1} + 0.2622x_{n-2} - 0.0009m_{n-1} + 0.0057m_{n-2}. \qquad (5.12)$$

The predicted output from this model gives a RMSE of 0.7991 with respect to the actual denoised outputs (Figure 5.4). Next, a wavelet-based least square identification technique is used. The model identified is

$$\hat{x}_n = 1.1459x_{n-1} - 0.1482x_{n-2} + 0.014m_{n-1} - 0.0127m_{n-2}. \qquad (5.13)$$

The RMSE using this model is 0.3442, which is around 57 % lower than that obtained using classical least square method. The comparison of the predicted values is shown in Figure 5.4. It is clear from this figure that wavelet-based identification method gives a better model in case of noisy raw input-output values in comparison to the classical least square method.

5.4.2 *Identification of the Liquid Level Process*

The liquid level process is described in section 4.2. This is a SISO system having input as the inlet water flow rate (in lit/min) and output as the liquid (water) level in the process tank (in mm). The water from the bottom tank is pumped to the process tank at the top by the centrifugal pump via the manual valve, rotameter, and the motorized valve. The outlet manual valve is kept open at one-quarter position. The inlet manual valve is kept fully opened. The flow rate in the inlet line is controlled by the motorized valve. The steady state is achieved by manually controlling the motorized valve to match the inlet flow rate with the outlet flow rate when desired level (80 mm) is reached. For a step change of 30 mm in the setpoint (water level in the tank) using an un-tuned dynamic matrix controller approximately sinusoidal inputs and outputs are generated from the experiment. The details of applying dynamic matrix controller are discussed in Chapter 6. These input-output data are used for identification of the liquid level process. A typical input-output raw data is shown in Figure 5.5. The denoised data using wavelet filter is also shown along with these data.

Similar procedure as that for identification of the heat exchanger process is adopted in this case also. Here two sets of input-output data have been used. One set has been used to identify the parameters and the other set for model validation. Figure 5.5 shows the raw noisy and denoised input-output data. Initially both first and second order model have been used to obtain the parameters by classical least-square identification. It is observed that the first order model gives a better prediction (RMSE = 2.3038) in comparison to the second order model (RMSE = 3.1285). The RMSE in case of first order model is about 35% lower than that in case of second order model (Figure 5.6). So, a first order model is assumed for the liquid level process. The model obtained in case of classical least square identification is

$$\hat{x}_n = 1.01231\, x_{n-1} + 0.77805\; m_{n-1}. \tag{5.14}$$

Next, the wavelet-based least-square identification method is used to obtain the model parameters. The model obtained is

$$\hat{x}_n = 0.998255\, x_{n-1} + 0.016822\; m_{n-1} \tag{5.15}$$

These two models are used to predict the output for a second set of input-output data. The prediction by the wavelet-based identification is better in terms of lower RMSE value. The RMSE is calculated with respect to the denoised output. The RMSE value in case of wavelet-based method is 1.6120, which is lower by around 30% from that of classical least square method (Figure 5.7).

5.5 Conclusions on Wavelet-based Identification of Chemial Processes

Both classical and wavelet-based least-square identification methods are applied to the example processes. For the heat exchanger process a second order model and for the liquid level process a first order model gives less RMSE while predicting the outputs from the inputs. The wavelet-based least-square identifications in both heat exchanger and liquid level processes are found to give better predictions than classical least-square identifications in term of lower RMSE while predicting 100 outputs from the inputs.

$$
\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{15} \\ x_{16} \end{bmatrix}
\xrightarrow{\underline{\underline{W}}}
\begin{bmatrix} s_1 \\ d_1 \\ s_2 \\ d_2 \\ s_3 \\ d_3 \\ s_4 \\ d_4 \\ s_5 \\ d_5 \\ s_6 \\ d_6 \\ s_7 \\ d_7 \\ s_8 \\ d_8 \end{bmatrix}
\xrightarrow{\text{Permute}}
\begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \\ \underline{s_8} \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \\ d_7 \\ d_8 \end{bmatrix}
\xrightarrow{\underline{\underline{W}}}
\begin{bmatrix} ss_1 \\ ds_1 \\ ss_2 \\ ds_2 \\ ss_3 \\ ds_3 \\ ss_4 \\ \underline{ds_4} \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \\ d_7 \\ d_8 \end{bmatrix}
\xrightarrow{\text{Permute}}
\begin{bmatrix} ss_1 \\ ss_2 \\ ss_3 \\ \underline{ss_4} \\ ds_1 \\ ds_2 \\ ds_3 \\ \underline{ds_4} \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \\ d_7 \\ d_8 \end{bmatrix}
\xrightarrow{\underline{\underline{W}}}
\begin{bmatrix} sss_1 \\ dss_1 \\ sss_2 \\ \underline{dss_2} \\ ds_1 \\ ds_2 \\ ds_3 \\ \underline{ds_4} \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \\ d_7 \\ d_8 \end{bmatrix}
\xrightarrow{\text{Permute}}
\begin{bmatrix} sss_1 \\ \underline{sss_2} \\ dss_1 \\ \underline{dss_2} \\ ds_1 \\ ds_2 \\ ds_3 \\ \underline{ds_4} \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \\ d_7 \\ d_8 \end{bmatrix}
$$

Figure 5.1 Discrete wavelet transform (DWT) of a discrete signal of 16 data points. The data vector is multiplied at the right side of the wavelet matrix $\underline{\underline{W}}$ to obtain the smooth and detailed coefficients. The smooth components are put together and again post-multiplied with the wavelet matrix. This procedure continues till finally two smooth components remain.

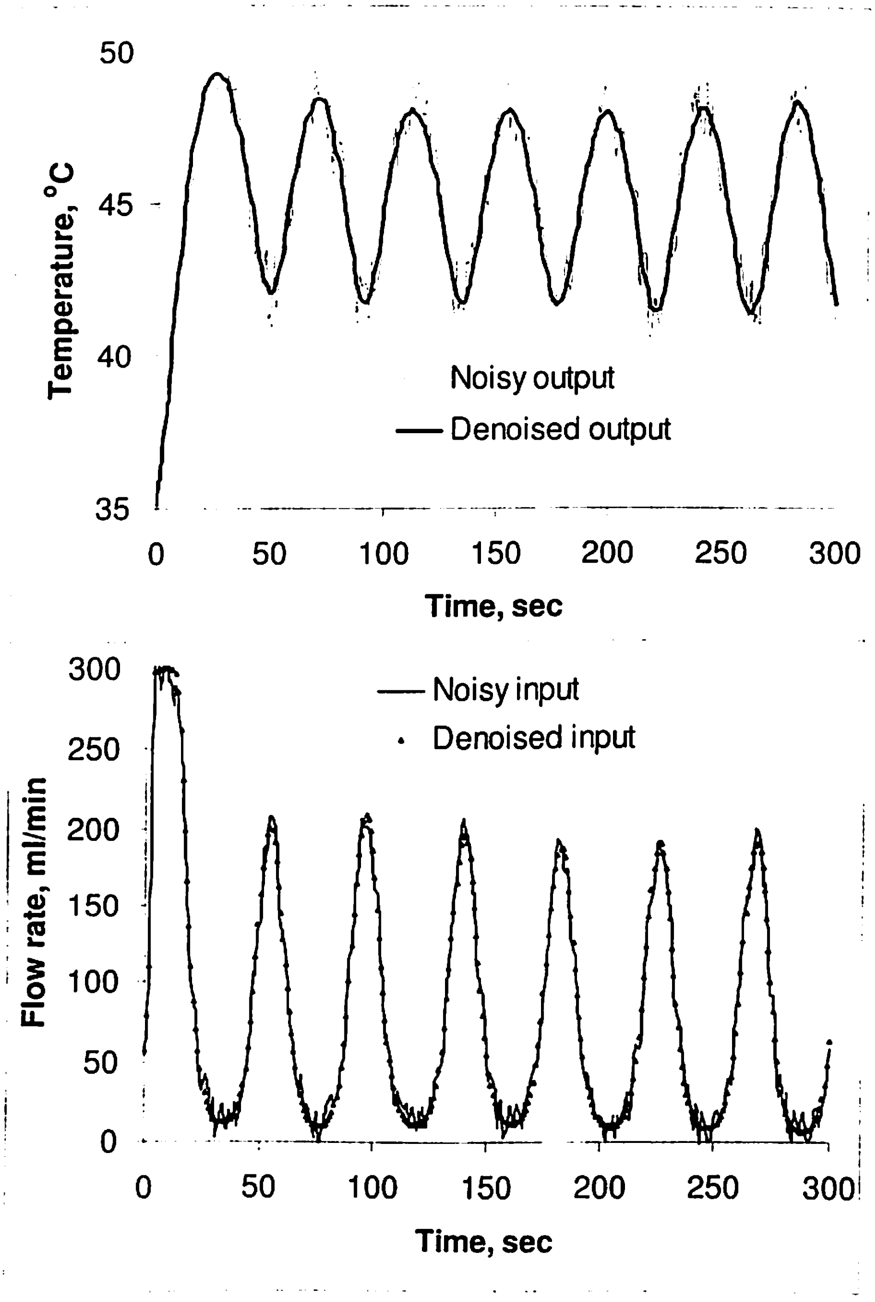


Figure 5.2 The input-output raw noisy data obtained from the heat exchanger setup and the corresponding denoised values using wavelet filter.

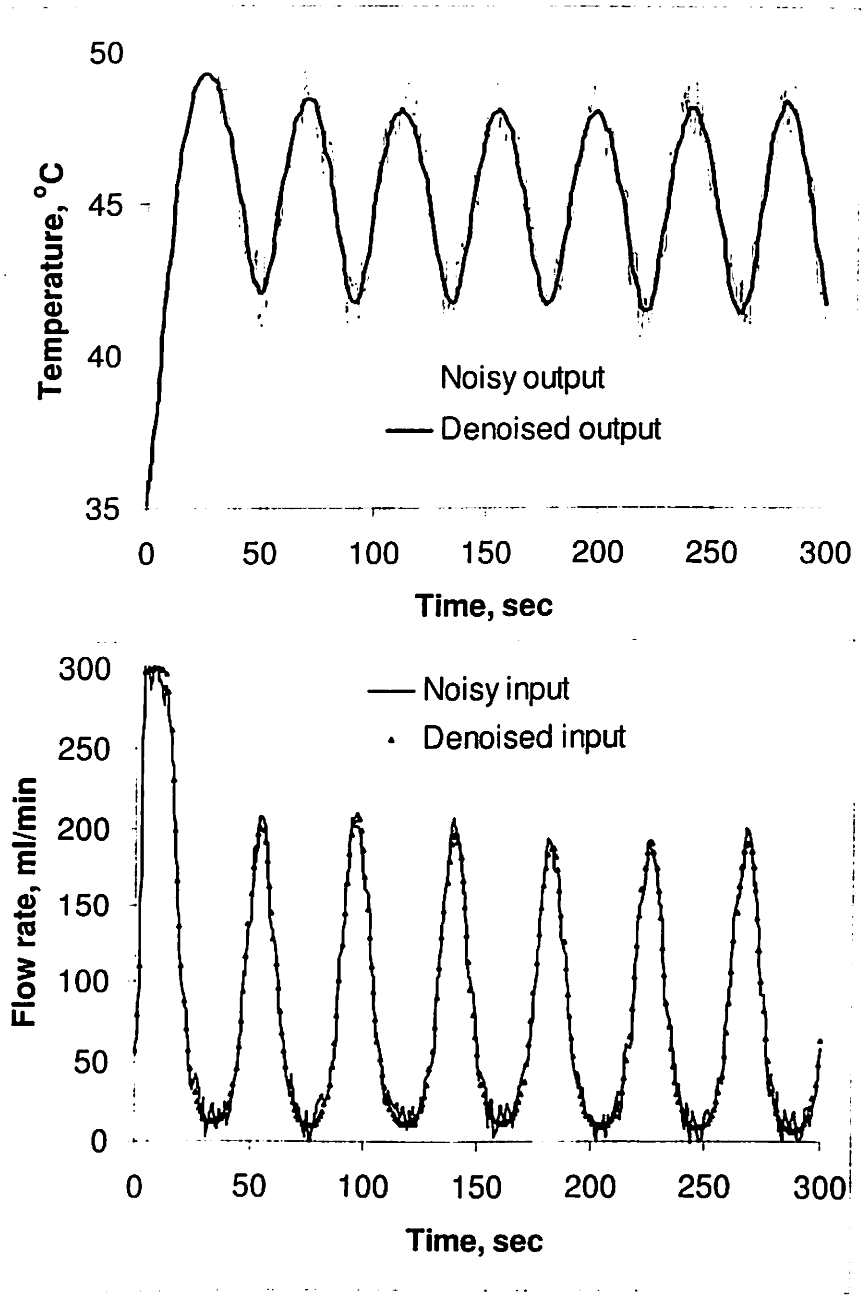


Figure 5.2 The input-output raw noisy data obtained from the heat exchanger setup and the corresponding denoised values using wavelet filter.

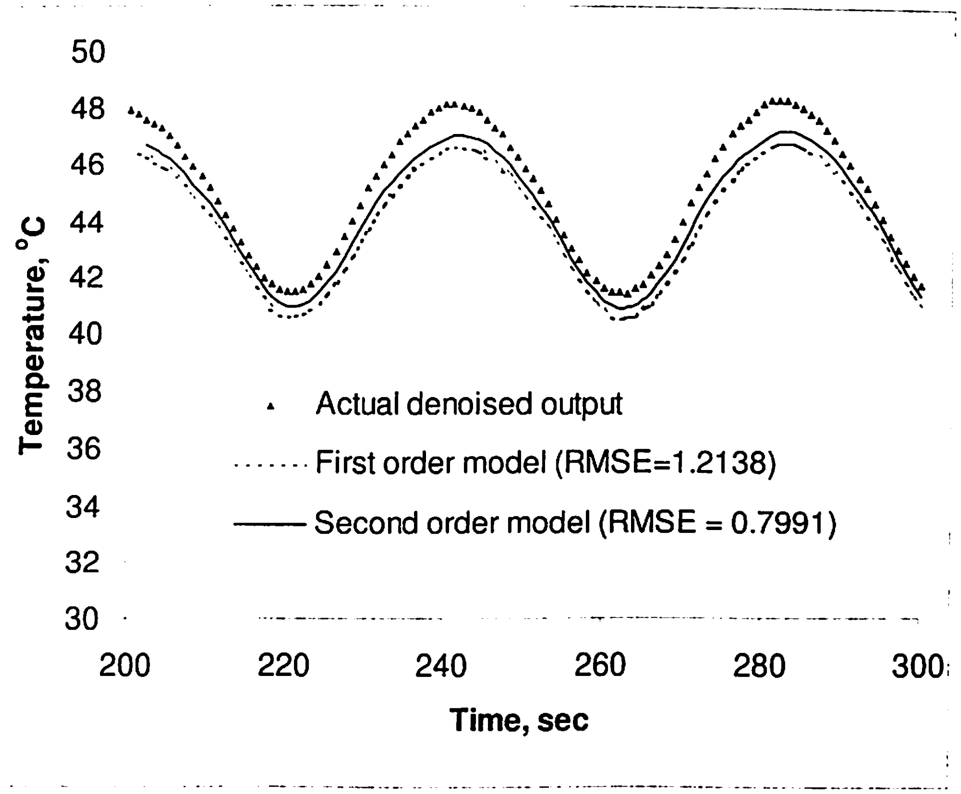


Figure 5.3 Comparison of the first and second order models for identification of the heat exchanger process.

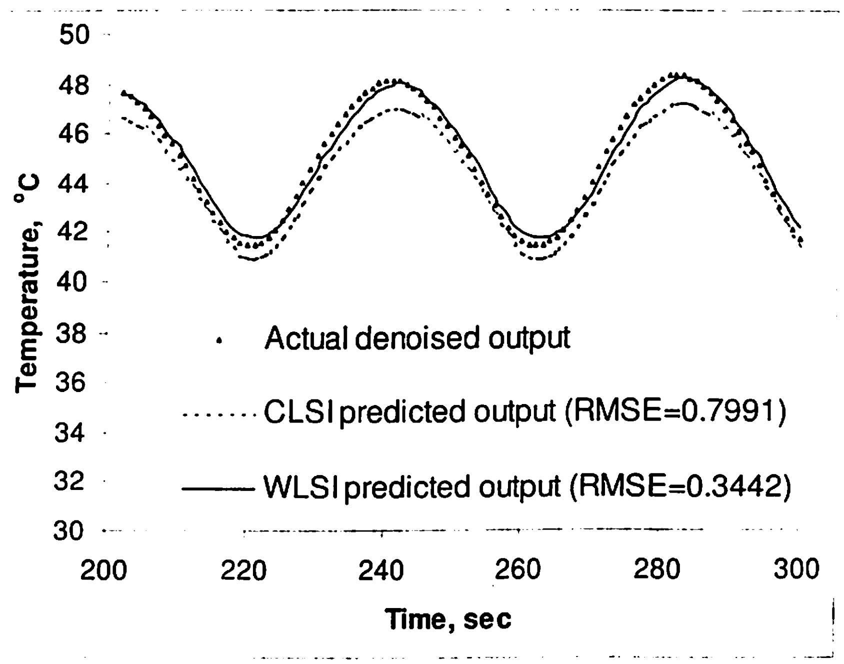


Figure 5.4 Comparison of the wavelet-based least square identification (WLSI) with the classical least square identification (CLSI) for the heat exchanger process.

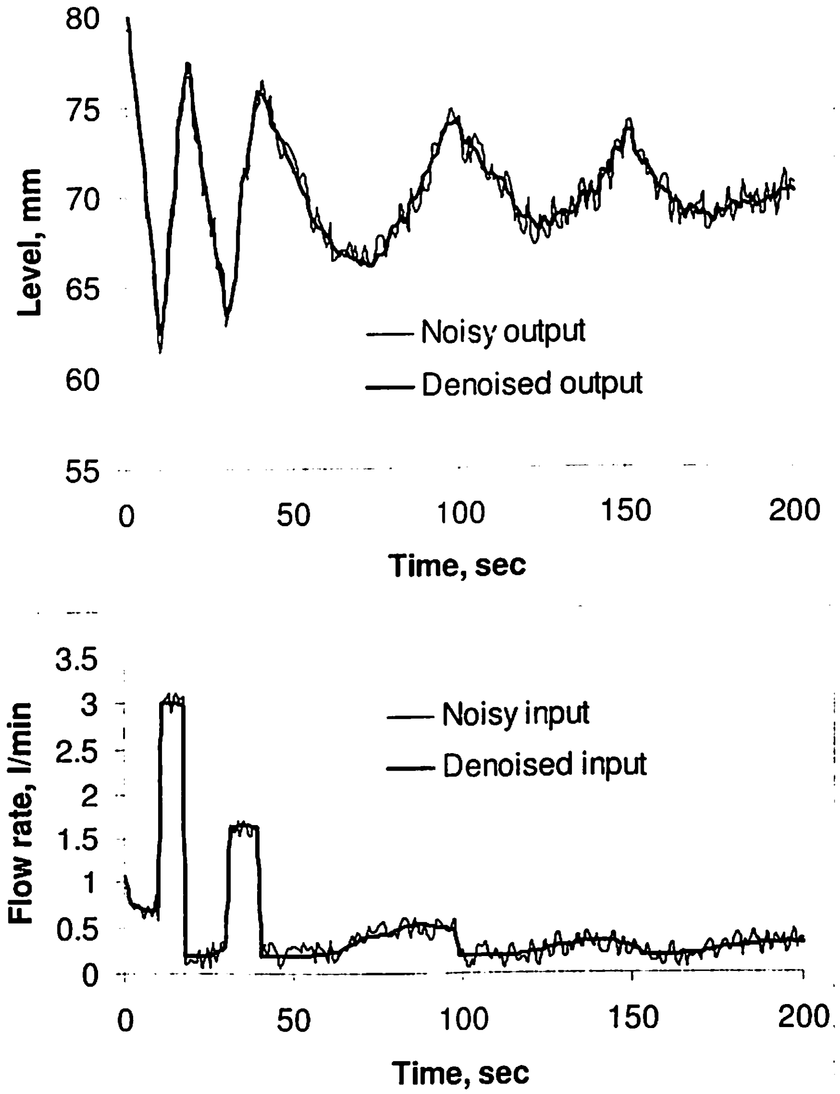


Figure 5.5 Input-output raw noisy data obtained from the liquid level process experiment and the corresponding denoised values using wavelet filter.

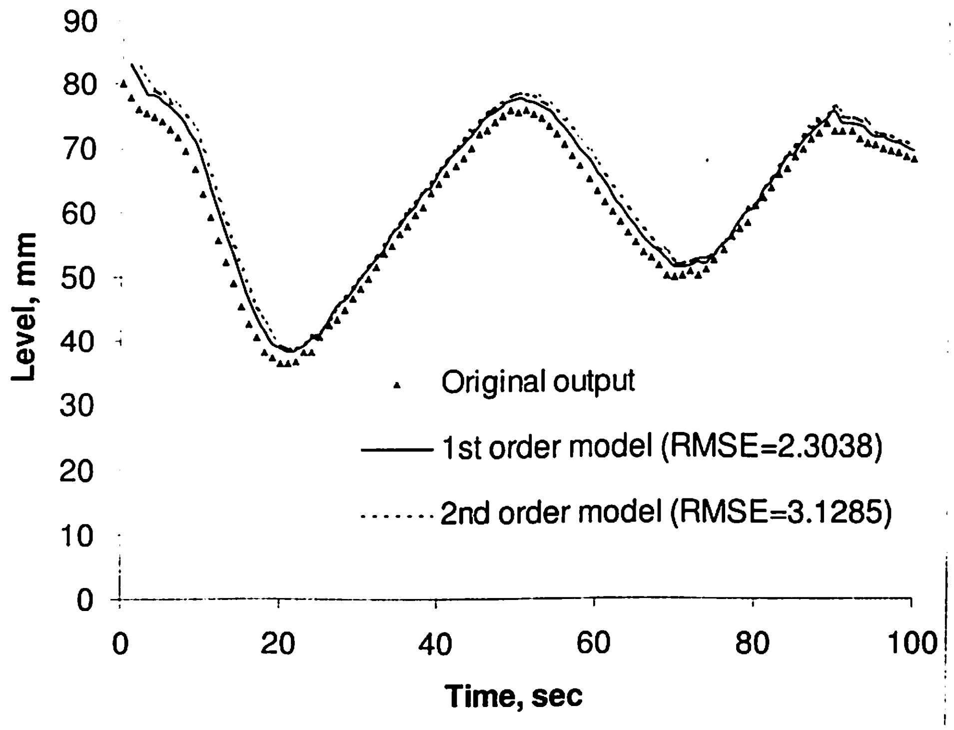


Figure 5.6 Comparison of the first order and second order models for identification of the liquid level process.

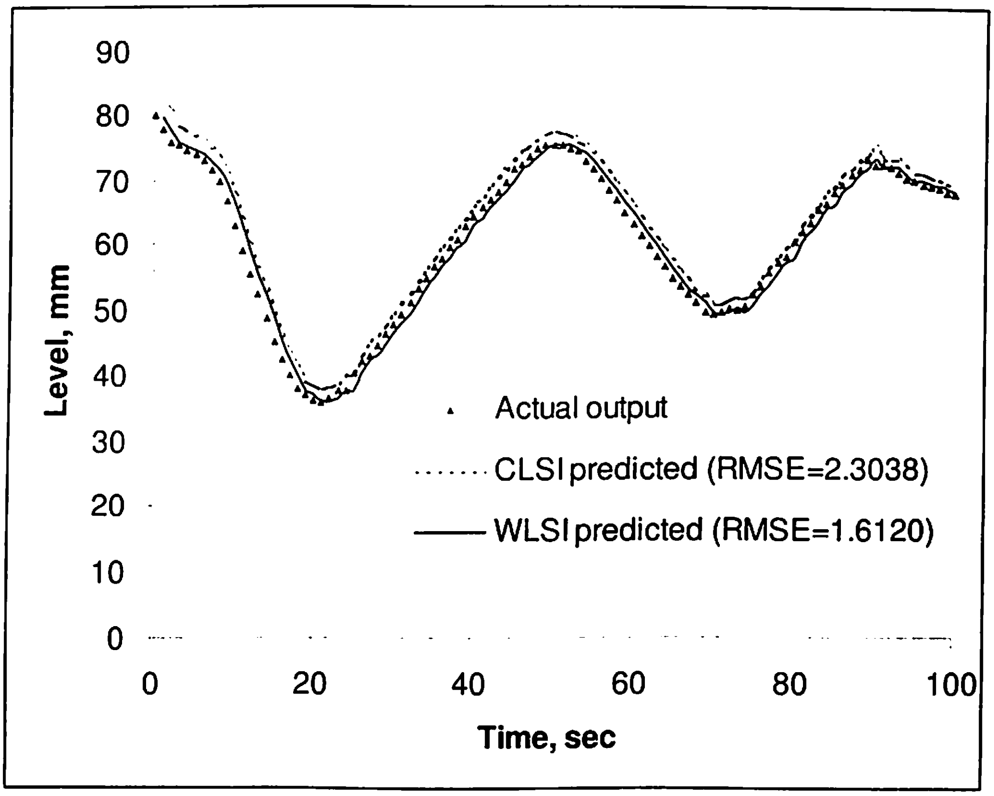


Figure 5.7 Comparison of the Wavelet-based least-square identification with the classical least-square identification for the liquid level process.

CHAPTER 6

WAVELET-BASED CONTROL OF CHEMICAL PROCESSES

This chapter discusses the brief theory behind the digital PID controller, Dynamic Matrix Controller (DMC), and the Wavelet-based Dynamic Matrix Controller (WDMC). All of these controllers are applied for temperature control in a Heat Exchanger Process and level control in a Liquid Level Process. The experimental results are discussed and finally the conclusions are drawn.

6.1 Digital PID Control

The continuous analog of a PID control action (Coughanowr, 1991; Stephanopoulos, 1993) can be represented by

$$c(t) = c_s + K_c\left[e(t) + \frac{1}{\tau_I}\int e(t)dt + \tau_D\frac{de(t)}{dt}\right] \tag{6.1}$$

Where e = error (i.e., set point - measured value of the output), c_s = controller's bias signal (i.e., its actuating signal when error, $e = 0$), K_c = proportional gain of the controller, τ_I = integral time constant and τ_D = derivative time constant.

The digital approximation of Equation (6.1) can be obtained by separately converting each term (proportional, integral, derivative) to an equivalent discrete-time one (Stephanopoulos, 1993).

Proportional term

In each sampling period (T) a sampled value of the process output enters the computer. Let $y(nT)$ be the sampled value of the n^{th} sampling instant. $y(nT)$ is compared to the set point value at the same instant and yields the value of the discrete-time error,

$$e(nT) = y_{SP}(nT) - y(nT)$$

Then the discrete-time control action produced by the proportional mode is

$$K_c e(nT)$$

Integral term

The control action produced by the integral mode is based on the integration of errors over a time period. Since the values of the errors are available on a discrete-time basis, the integral $e(t)dt$ can be approximated by numerical integration. Figure 6.1 shows the numerical evaluation of an integral using rectangular integration. It can be seen that

$$\int_0^t e(t)\,dt \approx T\sum_{k=0}^{n} e(kT)$$

Therefore, the integral mode control action is given by

$$\frac{K_c T}{\tau_I}\sum_{k=0}^{n} e(kT)$$

Derivative term

For the derivative mode action a numerical evaluation of the derivative de/dt is required. Figure 6.2 shows a first-order difference approximation of the derivative. Therefore,

$$K_c\tau_D\frac{de}{dt} \approx \frac{K_c\tau_D}{T}\{e(nT)-e[(n-1)T]\}.$$

Consequently, the control action of a digital PID controller is determined by the following discrete-time model:

$$c(nT)=c_s+K_c\left[e(nT)+\frac{T}{\tau_I}\sum_{k=0}^{n}e(kT)+\frac{\tau_D}{T}\{e(nT)-e[(n-1)]T\}\right]$$

Taking $nT = t_n$, the above equation can be expressed as

$$c(t_n)=c_s+K_c\left[e(t_n)+\frac{T}{\tau_I}\sum_{k=0}^{n}e(t_k)+\frac{\tau_D}{T}\left(e(t_n)-e(t_{n-1})\right)\right]. \tag{6.2}$$

The discrete controller output at $(n-1)^{th}$ sampling instance can be written as

$$c(t_{n-1})=c_s+K_c\left[e(t_{n-1})+\frac{T}{\tau_I}\sum_{k=0}^{n-1}e(t_k)+\frac{\tau_D}{T}\left(e(t_{n-1})-e(t_{n-2})\right)\right]. \tag{6.3}$$

Subtracting Equation 6.3 from Equation 6.2, the increment in controller output for one sampling interval is obtained as:

$$\Delta c=c(t_n)-c(t_{n-1})=K_c\left(1+\frac{T}{\tau_I}+\frac{\tau_D}{T}\right)e(t_n)-K_c\left(1+\frac{2\tau_D}{T}\right)e(t_{n-1})+K_c\left(\frac{\tau_D}{T}\right)e(t_{n-2}) \tag{6.4}$$

This form of equation (known as *velocity form*) is independent of the controller bias (c_s), protected against signal saturation (windup effect) and also protected against computer failure of the stepper motor position (Stephanopoulos, 1993). The values of only three running variables, the current error $e(t_n)$ and two previous error $e(t_{n-1})$ and $e(t_{n-2})$ are utilized for calculation. At starting, all the variables of previous time are initialized as zeroes. This form of PID control expression is rapidly executed in a computer with very less overhead of data storage capacity. A program in C language PID.C (Appendix I) is developed for implementing direct digital PID control.

6.2 Dynamic Matrix Control (DMC)

Dynamic Matrix Control (DMC) (Cutler and Ramaker, 1979, 1980; Ogunnaike, 1986; Maurath *et al.*, 1988; Luyben, 1990; Ogunnaike and Ray, 1994; Bozin and Austin, 1995; Saraf and Ganguly, 1997) is based on the following finite step response (or convolution) model of the process:

$$y(k) = \sum_{i=0}^{k} a(i)\Delta u(k-i) \tag{6.5}$$

where, the parameters $a(i)$ are the unit step response functions of the system with $a(0)=0$, $\Delta u(k) = u(k) - u(k-1)$, is the change in the input and $y(k)$ is the output of the system.

Let the current time instant be k, and control action has been taken at time k-1. In the absence of further control action, let

$$\hat{\mathbf{y}}^0(k) = [\hat{y}^0(k+1), \hat{y}^0(k+2), \ldots, \hat{y}^0(k+P)]^T$$

represent the values of the output variable over P-time step horizon (called prediction horizon).

The step response model (Equation 6.5) indicates that an arbitrary sequence of M control moves, $\Delta u(k)$, $\Delta u(k+1)$, ..., $\Delta u(k+M-1)$ will cause the system to change from $\hat{\mathbf{y}}^0(k)$ to a new state:

$$\hat{\mathbf{y}}(k+1) = [\hat{y}(k+1), \hat{y}(k+2), \ldots, \hat{y}(k+P)]^T$$

as given by the following equations, where the appended sequence, $w(k+1)$, $w(k+2)$, $w(k+3)$, ... $w(k+P)$, represents the collective effect of unmeasured disturbances on the output:

$$\hat{y}(k+1) = \hat{y}^0(k+1) + a(1)\Delta u(k) + w(k+1)$$

$$\hat{y}(k+2) = \hat{y}^0(k+2) + a(2)\Delta u(k) + a(1)\Delta u(k+1) + w(k+2)$$

$$\hat{y}(k+3) = \hat{y}^0(k+3) + a(3)\Delta u(k) + a(2)\Delta u(k+1) + a(1)\Delta u(k+2) + w(k+3)$$

$$\vdots$$

$$\hat{y}(k+M) = \hat{y}^0(k+M) + a(M)\Delta u(k) + a(M-1)\Delta u(k+1) + \ldots + a(1)\Delta u(k+M-1) \\ + w(k+M)$$

$$\hat{y}(k+M+1) = \hat{y}^0(k+M+1) + a(M+1)\Delta u(k) + a(M)\Delta u(k+1) + \ldots + a(2)\Delta u(k+M-1) \\ + w(k+M+1)$$

$$\vdots$$

$$\hat{y}(k+P) = \hat{y}^0(k+P) + a(P)\Delta u(k) + a(P-1)\Delta u(k+1) + \ldots + a(P-M+1)\Delta u(k+M-1) \\ + w(k+P)$$

In vector notation this may be written as

$$\hat{\mathbf{y}}(k+1) = \hat{\mathbf{y}}^0(k) + A\boldsymbol{\Delta u}(k) + w(k+1) \qquad (6.6)$$

where $w(k+1)$ is the vector of the collective effect of un-modeled disturbances; $\boldsymbol{\Delta u}(\boldsymbol{k})$ represents the M-dimensional vector of control moves given by

$$\boldsymbol{\Delta u}(k) = [\Delta u(k), \Delta u(k+1), \ldots, \Delta u(k+M-1)]^T$$

and the matrix **A** given by

$$\mathbf{A} = \begin{bmatrix} a(1) & 0 & 0 & \ldots & 0 \\ a(2) & a(1) & 0 & \ldots & 0 \\ a(3) & a(2) & 0 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ a(M) & a(M-1) & a(M-2) & \ldots & a(1) \\ a(M+1) & a(M) & a(M-1) & \ldots & a(2) \\ a(P) & a(P-1) & a(P-2) & \ldots & a(P-M+1) \end{bmatrix}_{P\times M} .$$

is called the system's "dynamic matrix". It is made up of M columns of system's step response function appropriately shifted down in order. For the output to reach the target, it has to follow a desired trajectory, called the reference trajectory (Figure 6.3), given as

$$\mathbf{y}^*(\boldsymbol{k}+\mathbf{1}) = [y^*(k+1), y^*(k+2), \ldots, y^*(k+P)]^T$$

So, the control moves $\boldsymbol{\Delta u}(\boldsymbol{k})$ are to be chosen such that

$$\hat{\mathbf{y}}(k+1) = \boldsymbol{y}^*(\boldsymbol{k}+\mathbf{1})$$

$$\text{or,} \quad \hat{\mathbf{y}}^0(k) + \mathbf{A\Delta u(k)} + w(k+1) = \boldsymbol{y}^*(\boldsymbol{k}+\mathbf{1}) \qquad (6.7)$$

or, $A\Delta u(k) = e(k+1)$ (6.8)

where $e(k+1)$ is the projected error vector, defined as

$$e(k+1) \equiv y^{*}(k+1) - [\hat{y}^{0}(k) + w(k+1)]. \quad (6.9)$$

The control objective (solution of Equation 6.8) is posed as a least-square optimization problem with a quadratic performance index (cost function) of the form

$$\underset{\Delta u}{Min}\ J = [A\Delta u - e(k+1)]^{T} Q\, [A\Delta u - e(k+1)] \quad (6.10)$$

where Q is a $P \times P$ positive definite weighting matrix.

In the unconstrained case, this minimization problem has a closed form solution, which represents the DMC control law

$$\Delta u(k) = (A^{T} QA)^{-1} A^{T} Q\, e(k+1) \quad (6.11)$$

Implementation of this control law results in excessive control action, especially when the control horizon is greater than 1 (Sridhar and Cooper, 1997). Hence, a quadratic penalty on the size of manipulated input moves is introduced into the DMC performance index. So, the modified performance index is given by

$$\underset{\Delta u}{Min}\ J = [A\Delta u - e(k+1)]^{T} Q\, [A\Delta u - e(k+1)] + \Delta u^{T} F \Delta u \quad (6.12)$$

where F is the weighting matrix of "move suppression parameter (MSP)".

In this case the resulting control action becomes

$$\Delta u(k) = (A^{T} QA + FI)^{-1} A^{T} Q\, e(k+1).$$

Figure 6.4 shows the block diagram for DMC. A program in C language DMC.C (Appendix I) is developed for implementing direct digital DMC control.

6.3 Wavelet-based Dynamic Matrix Control

A wavelet-based dynamic matrix controller (WDMC) is developed within the framework of DMC. Two changes are made in the structure of DMC to obtain WDMC: Firstly, the dynamic matrix is continuously updated using wavelet-based identification. Secondly, wavelet-based blocking and condensing (B & C) techniques are applied after converting the DMC problem into wavelet domain. The second modification reduces the size of the dynamic matrix without deteriorating the performance (Palavajjhala *et al.*, 1994). The block diagram for WDMC is shown in Figure 6.5. The wavelet-based identification is already discussed in the previous chapter. Wavelet-based B & C techniques are discussed here.

6.3.1 *Wavelet Transformation of the DMC Problem*

The performance index for DMC given by Equation 6.12 can also be written as

$$\min_{\Delta u} J = [e(k+1) - \mathrm{A}\Delta u]^T \mathrm{Q}\, [e(k+1) - \mathrm{A}\Delta u] + \Delta u^T F \Delta u \tag{6.13}$$

With the help of Equations 6.9 and 6.10, this can further be written as

$$\min_{\Delta u} J = [y^*(k+1) - \hat{y}(k+1)]^T \mathrm{Q}\, [y^*(k+1) - \hat{y}(k+1)] + \Delta u^T F \Delta u \tag{6.14}$$

Using l_2 norm the above equation can be represented as

$$\min_{\Delta u} J = \left\| Q(y^*(k+1) - \hat{y}(k+1)) \right\|_2^2 + \left\| F\Delta u(k) \right\|_2^2 . \tag{6.15}$$

The DMC objective function in time domain given by Equation 6.15 is to be converted into wavelet domain. The prediction horizon P and control horizon M are to be chosen such that

$$P = 2^{\hat{P}} \text{ and } M = 2^{\hat{M}} \text{ for } \hat{P}, \hat{M} \in N.$$

Let $U_c(k,t)$ be the continuous future input vector defined as follows:

$$U_c(k,t) = \begin{cases} 0 & \text{for } t < 0 \\ u(k+l) - u(k-l) & \text{for } l \le t < l+1,\ 0 \le l \le 2^{\hat{M}} - 1 \\ u(k+2^{\hat{M}}) - u(k-1) & \text{for } t \ge 2^{\hat{M}} \end{cases}$$

where $u(k-1)$ is the input implemented at the $(k-1)^{\text{th}}$ step.

The basis functions $\{\phi_{0,k}\}_{k=0,\cdots,2^{\hat{M}}-1}$ along with $\phi_{2^{\hat{M}},s}$, a unit step function starting at time $2^{\hat{M}}$, as a basis set, any $U_c(k,t)$ can be expressed as a linear combination of these $2^{\hat{M}}+1$ basis functions. The vector representation of $U_c(k,t)$ in terms of the above basis (denoted as $U(k)$) is as follows:

$$U(k) = \begin{bmatrix} u(k) - u(k-1) \\ \vdots \\ u(k+2^{\hat{M}}) - u(k-1) \end{bmatrix}$$

$\Delta u(k)$ and $U(k)$ are related as $\mathbf{\Delta u}(k) = \Pi\, U(k)$ where

$$\Pi = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ -1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & -1 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \ddots & -1 & 1 & 0 \\ 0 & 0 & 0 & \cdots & -1 & 1 \end{bmatrix}$$

Defining the output tracking error, $E(k) = y^{*}(k+1) - \hat{y}(k+1)$ and replacing $\Delta \boldsymbol{u}(k)$ with $\Pi\, U(k)$, we can rewrite the objective function as:

$$\underset{\Delta u(k)}{Min}\, J = \left\| QE(k) \right\|_2^2 + \left\| F\Pi U(k) \right\|_2^2 \tag{6.16}$$

The transformation of $U(k)$ from the standard pulse basis into an orthonormal wavelet basis can be performed by multiplying it with an appropriate wavelet matrix (Lee *et al.*, 1992). The continuous future input vector can be projected onto the wavelet basis as follows:

$$U_c(k,t) = \sum_{l=0}^{2^{\hat{M}}-1} (u(k+l) - u(k-1))\phi_{0,l} + (u(k+2^{\hat{M}}) - u(k-1))\phi_{2^{\hat{M}},s}$$

$$= u^{*}_{\hat{M},+}(k)\phi_{\hat{M},0} + \sum_{l=1}^{\hat{M}} \sum_{j=0}^{2^{\hat{M}-l}-1} u^{*}_{l,j}(k)\psi_{l,j} + u^{*}_{f}(k)\phi_{2^{\hat{M}},s} \tag{6.17}$$

Defining

$$U^{*}(k) = [u^{*}_{\hat{M},+}(k), u^{*}_{\hat{M},0}(k), u^{*}_{\hat{M}-1,1}(k), \cdots, u^{*}_{1,2^{\hat{M}-1}-2}(k), u^{*}_{1,2^{\hat{M}-1}-1}(k), u^{*}_{f}(k)]^{T} \tag{6.18}$$

$U(k)$ and $U^{*}(k)$ can be related through a unitary transformation:

$$U^{*}(k) = V_u U(k); \qquad V_u = diag\left[Ediag\{\underbrace{V,\cdots,V}_{n_u}\}, I_{n_u} \right]$$

V is a $2^{\hat{M}} \times 2^{\hat{M}}$ orthogonal matrix that performs a change of basis from the pulse basis to the wavelet basis and n_u is the number of elements in the vector $U^{*}(k)$. The notation "Ediag" refers to element-by-element diagonalization. For example,

$$Ediag\left\{\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}\right\} = \begin{bmatrix} \begin{bmatrix} a_{11} & 0 \\ 0 & b_{11} \end{bmatrix} & \begin{bmatrix} a_{12} & 0 \\ 0 & b_{12} \end{bmatrix} \\ \begin{bmatrix} a_{21} & 0 \\ 0 & b_{21} \end{bmatrix} & \begin{bmatrix} a_{22} & 0 \\ 0 & b_{22} \end{bmatrix} \end{bmatrix}$$

Similarly, the same wavelet transformation can be applied to the first 2^P elements of $E(k)$ and we get

$$E^*(k) = W_y E(k)$$

where $E^*(k)$ is the coefficient vector while expressing $E(k)$ with wavelet basis and W_y is an orthogonal matrix with the following structure:

$$W_y = diag\left[Ediag\{\overbrace{W,\cdots,W}^{n_y}\}, I_{n_x} \right]$$

W is a $2^P \times 2^P$ orthogonal matrix representing a change of basis from the pulse basis to the wavelet basis. The notation n_y refers to the number of elements in the vector $E^*(k)$.

The control objective can now be expressed in terms of the transformed variables as follows:

$$\underset{U^*(k)}{Min}\, J = \left\| Q W_y^T E^*(k) \right\|_2^2 + \left\| F\Pi V_u^T U^*(k) \right\|_2^2 \tag{6.19}$$

6.3.2 *Blocking and Condensing in Wavelet Domain*

In blocking, the dimension of the manipulated variable change vector (**Δu**) is reduced by constraining it to a lower dimensional subspace, whereas in condensing, the dimension of the future output tracking error vector (***E***) is reduced. In other words, the control horizon (*M*) and prediction horizon (*P*) are reduced to lower values by blocking and condensing (B & C).

The key to successful blocking and condensation lies in the proper choice of blocking projection matrix (P_B) and condensing projection matrix (P_C). Blocking and condensing design involves the proper selection of these projection matrices so that the transformed optimisation problem does not degrade controller performance significantly. Researchers (Lee *et al.*, 1992; Palavajjhala *et al.*, 1994) have shown the superiority of wavelet-domain B & C over the time-domain approach.

In Equation 6.17, the wavelet dilation and translation parameters j and l indicate the local position in time and frequency respectively. As we move from $l = 1$ to $l = \hat{M}$, the frequency decreases. Hence, the wavelet decomposition provides a convenient way to choose different control horizons for different frequency ranges. In order to implement this idea of frequency-varying horizon length, for each l, we can retain the first m_l elements of the l^{th} level wavelets and eliminate the rest. m_l is chosen according to the horizon length (counted in terms of the time unit of 2^l, which is the characteristic time span of an l^{th} level wavelet) appropriate for the frequency range corresponding to the l^{th} level wavelets $\psi_{l,k}$. Thus, the "blocked" manipulated vector can be expressed as:

$$U_B^{\bullet}(k) = P_B U^{\bullet}(k) \Rightarrow U^{\bullet}(k) = P_B^T U_B^{\bullet}(k)$$

where

$$U_B^{\bullet}(k) = [u_{\hat{M},+}^{\bullet}, u_{\hat{M},0}^{\bullet}, \{u_{\hat{M}-1,j}^{\bullet}\}_{j=0,m_{\hat{M}-1}}, \cdots, \{u_{1,j}^{\bullet}\}_{j=0,m_1}, u_f^{\bullet}]^T$$

and P_B is the blocking projection matrix derived from an identity matrix by eliminating the rows corresponding to the elements of $U^{\bullet}(k)$ that are to be blocked. A similar method can be applied in condensing the output vectors. The transformed future error vector $E^{\bullet}(k)$ is condensed as follows:

$$E_C^{\bullet}(k) = P_C E^{\bullet}(k) \Rightarrow E^{\bullet}(k) = P_C^T E_B^{\bullet}(k)$$

Here P_C is the condensing projection matrix, a fat matrix derived from an identity matrix by eliminating the rows corresponding to the outputs to be "condensed."

6.3.2.1 *Design Procedure for Blocking and Condensing*

Palavajjhala *et al.* (1994) developed a new blocking and condensing design methodology using information theory and sensitivity analysis. Design parameters d_u and d_y are introduced to select the blocking and condensing intervals. The design parameters specify the permissible error that can be introduced by approximation. The design involves following steps:

1. Assume d_u and d_y.
2. Select the condensing intervals using Haar wavelet packet decomposition and d_y.

3. Select blocking intervals using Haar wavelet packet decomposition and d_u.
4. Find the influence of the approximation on the first manipulated variable change. Update d_u if necessary and repeat step 3.
5. Compare closed loop performance, robustness and stability before and after "blocking" and "condensing" is used.
6. Update parameters d_u and d_y, if necessary repeat step 2 to step 5.

The residual error is used as the information criteria in this work. Residual error is defined as the l_2 norm squared of the difference between a sequence and its approximation, i.e. it is the energy loss in the approximation. Any approximation with an energy loss in approximation satisfying the following condition is considered admissible.

Energy loss in approximation $\leq \delta$ (energy of the sequence /length of the sequence)

where $\delta = d_y$ for selecting condensing intervals and $\delta = d_u$ for selecting blocking interval. Simulation studies (Palavajjhala *et al.*, 1994) suggest that a value of 0.05 of δ gives a good starting point.

A computer program in C language WDMC.C (Appendix I) is developed for implementing the Wavelet-based dynamic matrix control.

6.4 Results and Discussions

The developed Wavelet-based Dynamic Matrix Controller (WDMC) is implemented in two commonly used Chemical Engineering processes, namely the Heat Exchanger Process and the Liquid Level Process.

6.4.1 *Control of the Heat Exchanger Process*

The heat exchanger unit described in Section 4.1 is used for experimentation. The hot water sump is used with distilled water prior to any experiment. The heater and stirrer in the tank are connected to one of the 220V power supply sockets in the electrical console. In order to measure the temperature of water in the tank, the thermocouple in the tank is connected to the voltmeter. The voltmeter reading is calibrated as 0 to 1V for 0 to 100°C. Initially it takes about 30 minutes for the water in the tank to heat up to 80°C. During experiments, both heater and stirrer remain functional and a thermostat fitted in the tank maintains the temperature of the water around 80°C (80 ± 3°C). The manually operated valve in the hot water line is fully opened when water temperature reaches steady state.

The motorized valve is also partially opened. To operate the motorized valve, the "Metered-output" module in the electrical console is connected to the motor positioner and the same is then connected to the motor. By adjusting the knob in the analog "metered-output" with in 4-20mA, the motorized valve can be adjusted to any desired position. It is important to make the connection between the motor positioner to the motor, in order specified above. The open circuit voltage of motorized valve is very high and it tends to jam the valve by opening it beyond the full range if connections are not made in proper sequence.

The cold water is fed to the heat exchanger by partially opening the manual valve. The flow rate of the cold water is generally kept at 200 cc/min. The pump is now switched on, to circulate hot water through the heat exchanger. Due to heat loss to the metallic and to the cold fluid, the temperature of the hot water in the sump tank decreases. It takes about 10 min. to bring back the temperature of the hot water to 80°C. The set up is now ready for closed-loop study.

In the regulatory control experiments, the inlet flow rate of the cold water is used as a disturbance. Whereas in the servo control experiment, the set point of the outlet cold water is changed keeping other inlet variables constant. Some unnoticed disturbances such as inlet temperature of cold water and heat loss enter into the process in both the control problems. The thermocouple placed in the cold-water output line is connected to the process controller via signal conditioner. The output from the process controller is connected to the motorized valve through the motor positioner. This forms a feedback control loop.

A direct digital controller (DDC) is established by interfacing the process directly to the PC Bus, through the DT2811 AD/DA card. This arrangement facilitates very high sampling rate and exploit the high computing power of the computer for software-based control. The circuit diagram is shown in Figure 6.6.

The control objective is to keep the outlet temperature of the cold water at a desired set value using hot water flow rate as the manipulated variable. The possible disturbances are cold water flow rate, cold water inlet temperature and hot water temperature. Cold water is supplied from the overhead tank (OHT) and its temperature is assumed to be constant.

For servo problem, the cold water is fixed at a constant flow rate. The variation in hot water temperature is assumed to be negligible.

The outlet temperature of the cold water is converted to digital value using DT2811 card and is accessed through the source code ADC.C (Appendix I). For digital control, the computer processes this digital value and sends the output as a digital value called controller output (CO) between 0 and 4, which is accessed through the DT2811 card using the source code DAC.C (Appendix I).

6.4.1.1 *Calibration of the Motorized Valve*

The final control element of the heat exchanger process is a motorized valve. A number of experiments are conducted to characterize the motorized valve. Figure 6.7 shows its characteristics as a plot of hot water flow rate versus controller output (CO). The CO is a digital value between 0 and 4, which is converted to corresponding analog voltage signal of range 0 to 1 volt by the source code DAC.C (Appendix I). The relationship between CO and voltage is shown in Figure 6.8. It can be seen from Figure 6.7 that the valve characteristics are nonlinear for its full range. This non-linearity of final control element contributes in the nature of the closed loop response in different regimes. There are distinctly three regimes: regime I (0-50 cc/min), regime II (50-150 cc/min) and regime III (150-280 cc/min). For simplicity, regime II is considered for developing the model of the system. The motorized valve takes current signal for its performance. Therefore, the analog voltage signal (0-1 V) obtained from computer as controller is converted to current signal (4-20mA) using electrical console (Figure 4.3). This is done by taking the voltage signal to the voltage input of the signal conditioning channel where there is no signal conditioning block present and taking out the current output and putting into the motor positioner. Finally the motor positioner and the motorized valve are connected using 5-pin socket. The relationship between analog voltage and analog current signal is shown in Figure 6.9. Figure 6.10 shows the relationship between the controller output and current.

6.4.1.2 *Modeling of the Heat Exchanger Process*

The open-loop transfer functions of the process for servo and regulator problems are obtained through Cohen and Coon Process Reaction Curve (PRC) method (Cohen and Coon, 1953; Stephanopoulos, 1993; Ogunnaike and Ray, 1994) using the source code

MANCON.C (Appendix I). For servo problem, the hot water flow rate is given a step change while keeping the cold water flow rate constant. In case of regulator problem, the hot water flow rate is kept constant and a step change is introduced in the cold water flow rate. The PRCs (Figure 6.11 for servo problem and Figure 6.12 for regulator problem) are obtained in both cases. The First-Order-Plus-Dead-Time (FOPDT) model of the process is obtained as

$$y(s)=\frac{0.134e^{-s}}{32s+1}u(s)-\frac{0.1285e^{-s}}{62s+1}d(s) \tag{6.20}$$

where y is the cold water outlet temperature in °C, u and d are hot and cold water flow rates in cc/min respectively. All these variables are in deviation forms. The FOPDT models obtained for both servo and regulator problems are in good agreement with the experimental values. This is shown in Figure 6.13 for servo problem and in Figure 6.14 for regulator problem. The standard deviation in case of the model for change in the input is 0.1205 and that in case of model for change in disturbance 0.0710.

6.4.1.3 *PID Control of the Heat Exchanger Process*

The source code PIDTC.C (Appendix I) is used for implementing direct digital PID control. The starting temperature of the cold-water outlet is kept at 35°C for all the experiments and a step change of 10°C is given to study the closed loop performance in case of servo problems.

The Cohen and Coon approach (Cohen and Coon, 1953; Stephanopoulos, 1993) is used to estimate the initial controller parameters. The experimental process reaction curve is generated for a step change of 150 cc/min in the hot water flow rate and is plotted in Figure 6.11. The Cohen-Coon's estimated parameters are found to be

$$K_c = 15\frac{\text{cc/min}}{^\circ\text{C}},\quad \tau_I = 6.94\,\text{sec},\ \tau_D = 0.8\,\text{sec}$$

The closed loop response using these parameters (Figure 6.15) is however not satisfactory as it is oscillatory in nature and it gives higher overshoot and integral of square error (ISE). Then the online tuning is conducted using Ziegler-Nichols techniques (Ziegler and Nichols, 1942; Luyben and Luyben, 1997) and the parameters obtained are 13 (cc/min)/°C, 5.5 sec and 0.88 sec for K_c, τ_I and τ_D respectively. A better performance in terms of lesser oscillations and lower ISE value was obtained (Figure 6.16). Finally the controller parameters are modified through online-trial-and-error (OLTE) method using minimum ISE criteria and the following optimum value of the parameters are obtained:

$$K_c = 10\frac{\text{cc/min}}{^{\circ}C},\ \tau_I = 6\,\text{sec},\ \tau_D = 1\,\text{sec}.$$

The closed loop response using these parameters is shown in Figure 6.17. This response is better in terms of lesser overshoot and ISE. The comparison of the performances of the PID controllers with different settings is shown in Figure 6.18. The parameters for comparison are given in Table 6.1. It can be seen from this table that as the K_c value decreases, the ISE also decreases. This is because the higher the value of K_c, more the number of oscillations in the response and higher is the overshoot (Stephanopoulos, 1993). It can also be seen from this table that as τ_I value increases, the rise time increases. This is because integral action (higher τ_I) makes the closed-loop response more sluggish or slow (Stephanopoulos, 1993). The rise time and the overshoot in case of Ziegler-Nichols and Cohen-Coon settings are almost same as there is not much difference between the values of K_c, τ_I, and τ_D. The PID controller obtained by online-trial-and-error tuning is found to give better response compared to other two PID controllers.

For regulator problem, a step disturbance of 100 cc/min is introduced in the cold water flow rate by rotating the manual valve in the cold water flow line. Figure 6.19 shows the closed-loop response and the corresponding hot water flow rate for a regulatory control with PID settings. The step disturbance is introduced at around 100th second and the response comes back to its set value after 135 seconds. The overshoot is 1.23°C and the ISE in a period of 600 seconds is found to be 2410. The hot water flow rate, which is the manipulated variable, settles at a higher value (hot water flow rate changes from 90 cc/min to 240 cc/min with an overshoot). This is to compensate for the higher cold water flow rate (which is changed from 100 cc/min to 200 cc/min). It is observed that the controller is able to bring back the process to original steady state. This can be called as a robust controller because this is able to reject a moderate disturbance while maintaining stability (Morari and Zafiriou, 1989).

6.4.1.4 *DMC of the Heat Exchanger Process*

The source code DMC.C (Appendix I) is used for implementing dynamic matrix control (DMC) of the heat exchanger process. The step response coefficients are obtained from the open-loop response of the process for a step change in the hot water flow rate. This is same as the process reaction curve of the process in case of servo problem and is shown in Figure 6.11.

The tuning of the dynamic matrix controller is done by online trial and error (OLTE) method. The tuning parameters in DMC controller are prediction horizon (NP), control horizon (NC), model or truncation horizon (NT) and move suppression parameter (MSP) and sampling time. The sampling time is kept at 1 second for ease of operation.

The upper values of prediction horizon, NP and control horizon, NC are restricted by the processing capacity of the computer. NC is typically set at 50% of NP (Luyben, 1990). It is observed that the computer used is not able to process a matrix bigger than 40x20 in size. So, the upper limits of NP and NC are 40 and 20 respectively. For wavelet-based control, the NP and NC values must be expressed as integer powers of 2. Therefore, the initial values of NP and NC are taken as 32 and 16 respectively. The initial value of NT is taken to be same as NC.

The same initial starting temperature of cold water outlet (35°C) and a step change of 10°C in the set point as used in PID control was maintained in this case.

Effect of MSP

Initially the DMC is tested off-line to find the range of MSP to be used in control. It is observed that the MSP value below 100 gives a large change in hot water flow rate (above 280 ml/min) and a value more than 2000 gives very little change in hot water flow rate (around 0 ml/min). So, MSP values are taken in the range 100 to 2000. The closed-loop performances of the DMC with different MSP values keeping other parameters constant are shown in Table 6.2. The performance of the DMC is judged through ISE between the set point and the response for a period of 300 seconds. It can be seen from Table 6.2 that the ISE initially decreases and then continuously increases. This is because at lower MSP, the control action is very fast and the oscillations are more. At higher MSP, the control action is suppressed and the response becomes sluggish. These two factors contribute for higher ISE. The minimum ISE is found when MSP value is 200. So the DMC is tuned to this MSP value.

Effect of NC

The effect of NC on the performance of DMC is shown in Table 6.3. It can be observed from this table that the ISE initially decreases and then increases. This is because NC is the number of steps of control moves through which the desired control action can be achieved. If NC value is smaller the controller has to take large steps in order to achieve the desired control action. So, the response becomes oscillatory and that increases the ISE. But as NC increases, the oscillations are lessened and ISE decreases. At higher NC, the number of steps for control moves becomes high, which forces the controller to become sluggish. That's how ISE increases again. The ISE is found to be minimum when NC is 16. The DMC is tuned for this value of NC.

Effect of NT

The effect of NT on the performance of DMC is presented in Table 6.4. It can be observed from this table that ISE initially decreases and then increases for increase in NT. This is because NT is the model horizon, which is responsible for the response due to past control actions. DMC is based on the convolution model of the process, which takes care of the effects due to past control actions through NT. If NT is less, less number of past control actions are considered for contributing to the actual response. So, the error calculated by DMC is more than the actual error. This drives the controller faster (i.e. rise time is less) and gives rise to higher overshoot and more oscillations. Thus ISE is more in case of low NT. But as NT is increased, number of past control actions considered for contributing to the actual response are more. In this case if the convolution model obtained at a particular operating point does not valid at a different operating point (due to non-linearity in the process), then the actual error will also increase. Again the controller will be very fast and will give higher overshoot and more oscillations, which eventually increases the ISE value. In the present case the process is found to be a nonlinear process (Figure 6.7) and the convolution model obtained is valid only in the middle regime (refer to Section 6.4.1.1 on "Calibration of the Motorized Valve"). The mismatch between process and model leads to higher ISE at high value of NT. The optimum NT is found to be 17. The DMC is tuned for this value of NT.

The closed-loop response of the DMC for servo control problem of the heat exchanger process (same as that in case of the PID controller) is shown in Figure 6.20. The regulatory control action of the DMC is shown in Figure 6.21.

6.4.1.5 *Wavelet-based Dynamic Matrix Control of the Heat Exchanger Process*

The tuned parameters for DMC, i.e., NP=32, NC=16, NT=17 and MSP=200 are used in case of wavelet controller. Maiti and Saraf (1995) suggested that for adaptive DMC the identification should be started at the sampling time (NI_{min}) at which the open loop response reaches 50 – 60% of its final steady state value and should be stopped the time (NI_{max}) when the open loop response reaches 90 – 95% of its final steady state value. The reason being at lower value (below NI_{min}) the identified model becomes worse than the existing model and hence the performance of the new controller becomes poorer. At higher value (above NI_{max}) the performance does not improve. In the present case this two limits are found to be 20 sec and 100 sec, respectively (Figure 6.11) and the identification is conducted at 20th, 25th, 30th, ..., 100th seconds using wavelet-based method. The blocking coefficients are obtained from Figure 6.22 and the condensing coefficients from Figure 6.11. The WDMC is tested on-line with different values of design parameters d_y and d_u. The effect of d_y on the performance of WDMC is shown in Table 6.5. It can be observed from this table that initially ISE decreases and gives a minimum value at $d_y = 0.02$ and then increases. This is because when d_y is lower, effective NP is higher, which makes the controller more sensitive and the oscillations are more. This leads to higher value of ISE. When d_y is higher, effective NP is lower, which forces the controller to take large steps. This again increases the ISE. The optimum value of 0.02 is tuned for d_y. Similarly the WDMC is tested for blocking design parameter d_u and the performances are presented in Table 6.6. This table shows that as d_u increases, ISE first decreases and then increases. At lower d_u, the effective NC is higher, which means more control moves are required to achieve the desired state. This makes the controller slow and thereby ISE increases. At higher d_u, the effective NC is lower and bigger control moves are required to achieve the desired state. This again increases ISE. So, the optimum d_u of 0.05 is taken.

Using the tuned parameters the performance obtained for WDMC is far better in comparison to that of DMC or PID control (Figure 6.23). Dasgupta and Gupta (1998) have implemented Fuzzy Logic based Controller (FLC) in the same experimental setup for identical conditions. Figure 6.24 shows the performances of different controllers for a step change 10°C (from 35°C to 45°C) in the setpoint. The comparison is shown in Table 6.7. It can be observed from this table that ISE is highest (1316) in case of PID controller and lowest (730) in case of WDMC. Though the FLC of Dasgupta and Gupta (1998) has lesser overshoot (0.2°C) compared to that (1.0°C) of WDMC, its ISE is higher (813) than that of WDMC (730). The DMC has very less rise time (16 sec), but overshoot is highest (2.89°C) and ISE is higher (817) than that of FLC and WDMC. For better performance it is required that rise time, overshoot and ISE should be as minimum as possible. WDMC is better in terms of least ISE. The rise time is not very high compared to that of DMC, which has the lowest value. Also the overshoot is not very high compared to that of FLC, which has the lowest overshoot. So, the overall performance of wavelet controller is better than PID, DMC and FLC.

For regulatory problem, a step change of 100 cc/min is given in the cold water flow rate and the response given by the WDMC is shown in Figure 6.25. The WDMC is found to be robust as it is able to reject the moderate disturbances while maintaining the stability. A comparison of the performances of PID, DMC and WC controllers for regulatory action and for same step change in the cold water flow rate is show in Figure 6.26. The performance parameters are presented in Table 6.8. From this table it can be observed that the wavelet controller gives best performances in terms of least ISE and moderate values of rise time and overshoot.

6.4.2 *Control of the Liquid Level Process*

The liquid level control experimental setup described in Section 4.2 is used. The circuit diagram for computer control of the liquid level process is shown in Figure 6.27. Initially PID control is established and then DMC and WDMC are developed. The experiments are planned in the light of prior experience in heat exchanger control. The sump tank is filled up with water from external source. In order to pump water into the process tank, both valves in the water flow line (manual as well as motorized) are opened. The procedure for opening the motorized valve with the help of the electrical console is already explained (Section 6.4.1). The pump is now started by connecting it to one of the 220 V power supply sockets in the electrical console. The range of the rotameter in the inlet line is 0-3 lit/min. However, when the valve is fully open, the flow is slightly more than 3 lit/min. The pump is kept on during the operation of the process and when the motorized valve is closed the water re-circulates back to the sump tank through the bypass line. The motorized valve has a closing error of about 0.2 lit/min. This closing error of the valve leads to overflow of the process tank through the overflow drainage (even if the motorized valve is closed) when the pump is on and also leads to back flow of water to the sump tank when pump is stopped.

The level of water in the process tank can be measured with a scale attached to the body of the tank. It can also be automatically recorded from the output of a float type level meter. The output from the level meter is fed to the signal conditioner and the signal conditioner voltage output is connected to the computer as digital process controller through IBM-Adapter and DT2811 AD/DC card. The controller output of 0 to 100% corresponds to actual level of 30 mm to 130 mm in the process tank. The level below 30 mm is not measurable. When water level in the tank reaches beyond 140 mm, it overflows through the overflow drainage pipeline. Apart from the overflow drainage, the process tank also has three drainage pipelines of different sizes, numbered 1,2 and 3, at the bottom of the tank. The drainages through these lines are proportional to the level of water in the process tank. Drainage line 1 is smallest in diameter and is operated by a solenoid valve to fully close or fully open position. The line 3 is largest in diameter and is operated by a quarter turn manual valve. The line 2 is of intermediate size and is also controlled by a solenoid valve. Some combination of drainage valves is always kept open during experiments because the motorized valve can only control the inflow and in the event of overshoot, it is not able to bring the level down. In the present case the manual valve is kept one quarter open. The solenoid valves are used to introduce disturbance in case of regulatory problem.

Both servo as well as regulatory control actions are studied with PID, DMC and WDMC controllers. For servo control problem, a step change of 30 mm is given in the setpoint when the level attains a steady value of 80 mm. For regulatory control problem, a pulse change of -20 mm is given in the liquid level when it reaches steady state at 80 mm.

6.4.2.1 *PID Control of the Liquid Level Process*

To control the water level in the tank, the source code PID.C (Appendix I) was modified for digital PID control of liquid level through computer. The controller output, which is coming from the computer via DT2811 DAC card as a voltage signal is converted to corresponding current signal by the open channel of the electrical console and is connected to the motor positioner.

Initially the tuning parameters are determined through Cohen-Coon method. The process reaction curve (Figure 6.28) is obtained by giving a step change of 50 cc/min in the inlet flow rate. The process model obtained is given by

$$\frac{y(s)}{u(s)}=\frac{9e^{-3s}}{90s+1} \tag{6.21}$$

where, y(s) is the output (level) in mm in deviation form and u(s) is the input flow rate in cc/min in deviation form. This FOPDT model of the liquid level process is in good agreement with the experimental values as it gives a standard deviation of 0.1064.The Cohen-Coon settings for the PID controller are obtained as K_c = 4.5 (cc/min)/mm, τ_I = 7 sec. and τ_D = 1 sec. The response of the controller with these settings is shown in Figure 6.29. The Ziegler-Nichols settings are found out to be K_c = 3 (cc/min)/mm, τ_I = 6 sec and τ_D = 1.5 sec and the response using these settings is shown in Figure 6.30. The settings obtained by online-trial-and-error method are K_c = 4 (cc/min)/mm, τ_I = 10 sec and τ_D = 1 sec. The response for these settings is shown in Figure 6.31. Figure 6.32 shows the comparison of the PID controllers with different settings. The performances of each PID controller are presented in Table 6.9. It can be observed from the table that the PID controller with settings obtained from OLTE method give better performance in terms of lesser overshoot and ISE value. The closed-loop response of this controller for regulatory control is shown in Figure 6.33.

6.4.2.2 *DMC of the Liquid Level Process*

The step response coefficients for DMC are obtained from Figure 6.28. The source code used for DMC in case of heat exchanger is modified for liquid level control. For servo control problems similar procedures are followed as that of the temperature control in the heat exchanger. The NP value in this case is kept at 16. The NC value is taken to be 8, i.e., 50% of NP (Luyben, 1990) and NT to be 9 from the relationship, NP = NT+NC-1

(Garcia and Morari, 1982; Cutler, 1983; Georgiou *et al.*, 1998). The DMC is tuned on-line for different values of MSP. The performance factors are presented in Table 6.10. It can be observed from this table that an MSP value of 10 gives minimum ISE. Therefore, MSP is set to this value. With these settings, the performance of DMC for servo and regulatory problems are shown in Figure 6.34 and Figure 6.35 respectively.

6.4.2.3 *Wavelet-based Control of the Liquid Level Process*

The wavelet-based controller is developed in the DMC framework. In DMC, the step response model of the process is used, whereas, in case of wavelet-based controller the model obtained by wavelet-based identification is used. The tuned parameters of DMC are taken for direct implementation. The other tuning parameters are the condensing design parameter, d_y and blocking design parameter, d_u. Using on-line tuning (Table 6.11 and Table 6.12) these values are found to be $d_y = d_u = 0.05$. The performance of WDMC for servo problem is shown in Figure 6.36.

The performances of PID, DMC and WDMC controllers for servo problem are shown in Figure 6.37. The comparative study in terms of rise time, overshoot and ISE are presented in Table 6.13. It can be seen that DMC is the best in terms of fastest rise time. But it has higher overshoot and hence higher ISE. The WDMC has higher rise time, but has lesser overshoot and least ISE.

The robustness of WDMC is studied through regulatory controls. The closed-loop response of the liquid level process for regulatory problem using WDMC is shown in Figure 6.38. The WDMC is found to be robust as it is able to reject the moderate disturbances while maintaining the stability. Dasgupta (1999, 2004) has implemented ANN-based controller for the same experimental setup for identical conditions. The performances of PID, DMC, WDMC and ANN controllers for regulatory problem are shown in Figure 6.39. The comparative study in terms of rise time, overshoot and ISE are presented in Table 6.14. It can be seen from this table that the ANN controller of Dasgupta (2004) gives least overshoot, but its ISE is higher than that of the WDMC. The PID controller gives lowest performance and better than this is the DMC. In case of WDMC the rise time and ISE are lowest.

For better performance, a controller should have less rise time and overshoot. But it is observed that when the rise time is less, the overshoot becomes more and vice versa. So, the performance of the controller cannot be judged either from rise time or overshoot alone. When we ask for less rise time and less overshoot, actually we seek less ISE. So, it is obvious that the controller, which gives less ISE, is the best. It can be seen from Table 6.7 and Table 6.14 that the wavelet-based controller is the best in this regard among PID, DMC, FLC and ANN controllers.

The processing times (CPU times) required by the DMC and WDMC for control of the heat exchanger process are compared. The values obtained for first 50 samples are shown in Figure 6.40. It can be seen from this figure that the processing time fluctuates between an upper and a lower limit. Both the lower and upper limits of CPU time for WDMC are lower than those of DMC. So, it is understood that WDMC takes lesser CPU time than DMC. It is because WDMC uses a reduced size dynamic matrix in comparison to DMC by the application of the B & C techniques. The performance of the WDMC does not deteriorate when the dynamic matrix is reduced. This is due to the fact that the B & C techniques enable the controller to take only the required steps.

6.5 Conclusions on Wavelet-based Control of Chemical Processes

The FOPDT models for the heat exchanger process and the liquid level processes are in good agreement with the experimental results as the standard deviations are found to be in the order of 10^{-1}. The PID, DMC and WDMC controllers are developed and successfully implemented in the heat exchanger- and the liquid level processes. The PID controller with the OLTE settings give better performance than the PID controllers with C-C and Z-N settings. The DMC gives better performance than the PID controller in both the processes. The WDMC gives even better performance than the DMC. The CPU time required by WDMC is lower than that of DMC. The developed PID, DMC and WDMC controllers are found to be robust as they are able to reject the moderate disturbances while maintaining the stability in case of regulatory controls. It is observed that the developed WDMC controller outperforms the PID, DMC, FLC and the ANN-based controllers in terms of least ISE in both the processes.

Table 6.1 Comparison of the performances of different PID controllers for a step change of 10 °C (set point change from 35°C to 45°C) in the heat exchanger process.

PID Controllers (Settings)	**Controller Parameters**			**Rise time (s)**	**Overshoot (°C)**	**ISE (for 300 s)**
	K_c (cc/min)/°C	**τ_I (s)**	**τ_D (s)**			
PID (C-C)	15	5	0.8	20	6.94	2751
PID (Z-N)	13	5.5	0.88	21	7.3	2218
PID (OLTE)	10	6	1	27	2.5	1316

Table 6.2 Effect of MSP on DMC with NP = 32, NC = 16 and NT = 16 in the heat exchanger process for a step change of 10°C in the setpoint.

MSP	**Rise time (s)**	**Overshoot (°C)**	**ISE (for 300 s)**
100	21	3.75	1232
200	17	2.84	920
300	19	3.2	1286
400	19	4.92	1356
500	23	4.2	1416
600	21	3.75	1468
1000	21	4.09	1514
1500	26	3.9	1644
2000	23	4.34	2085

Table 6.3 Effect of NC on DMC with NP = 32, NT = 16 and MSP = 200 in the heat exchanger process for a step change of 10°C in the setpoint.

NC	**Rise time (s)**	**Overshoot (°C)**	**ISE (for 300 s)**
8	20	4.8	2067
12	18	3.5	1360
16	17	2.84	920
20	19	2.72	947
24	21	3.5	1081

Table 6.4 Effect of NT on DMC with NP = 32, NC = 16 and MSP = 200 in the heat exchanger process for a step change of 10°C in the setpoint.

NT	Rise time (s)	Overshoot (°C)	ISE (for 300 s)
15	12	4.23	2093
16	17	2.84	920
17	16	2.89	817
18	13	4.33	2041

Table 6.5 Effect of condensing design parameter d_y on WDMC with NP = 32, NC = 16, NT = 17, MSP = 200 and $d_u = 0.1$ in the heat exchanger process for a step change of 10°C in the setpoint.

d_y	Rise time (s)	Overshoot (°C)	ISE (for 300 s)
0.005	27	3.73	1407
0.02	25	3.43	1232
0.05	22	3.90	1435

Table 6.6 Effect of blocking design parameter d_u on WDMC with NP = 32, NC = 16, NT = 17, MSP = 200 and $d_y = 0.2$ in the heat exchanger process for a step change of 10°C in the setpoint.

d_u	Rise time (s)	Overshoot (°C)	ISE (for 300 s)
0.02	25	3.41	1255
0.05	19	1.0	730
0.10	25	3.43	1232

Table 6.7 Comparison of the performances of different controllers for a step change of 10°C in the set point (set point change from 35°C to 45°C) for the heat exchanger process.

Controllers	Rise time (s)	Overshoot (°C)	ISE (for 300 s)
PID	27	2.5	1316
DMC	16	2.89	817
WDMC	19	1.0	730
FLC (Dasgupta and Gupta, 1998)	32	0.2	813

Table 6.8 Comparison of the performances of different controllers for a step change of 100 cc/min in the cold water flow rate for the heat exchanger process (regulatory problem).

Controllers	Rise time (s)	Overshoot (°C)	ISE (for 600 s)
PID	135	1.23	2410
DMC	100	0.6	1612
WDMC	102	0.8	736

Table 6.9 Comparison of the performances of PID controllers with different settings in the liquid level process for a step change of 30 mm in the level.

PID Controllers (Settings)	Controller Parameters			Rise time (s)	Overshoot (mm)	ISE (for 300 s)
	K_c (cc/min)/mm	τ_I (s)	τ_D (s)			
PID (C-C)	4.5	7.0	1.0	26	5.07	7806
PID (Z-N)	3.0	6.0	1.5	26	4.15	7677
PID (OLTE)	4.0	10.0	1.0	26	3.7	6580

Table 6.10 Effect of MSP on DMC with NP = 16, NC = 8 and NT = 9 in the liquid level process for a step change of 30 mm in the level.

MSP	Rise time (s)	Overshoot (mm)	ISE (for 300 s)
0.01	12	19	15120
0.05	12	18.1	10905
0.10	15	13.5	12783
10	18	4.5	5725
100	29	10.35	16202

Table 6.11 Effect of condensing design parameter d_y on WDMC with NP = 16, NC = 8, NT = 9, MSP = 10 and d_u = 0.05 in the liquid level process for a step change of 30 mm in the level.

d_y	Rise time (s)	Overshoot (mm)	ISE (for 300 s)
0.02	13	9.15	6374
0.05	24	1.90	5140
0.10	15	10.5	10767

Table 6.12 Effect of blocking design parameter d_u on WDMC with NP = 16, NC = 8, NT = 9, MSP = 10 and d_y = 0.05 in the liquid level process for a step change of 30 mm in the level.

d_u	Rise time (s)	Overshoot (mm)	ISE (for 300 s)
0.02	29	5.92	11253
0.05	24	1.90	5140
0.10	12	9.25	6139

Table 6.13 Comparison of the performances of different controllers for the liquid level process for a step change of 30 mm (from 80 mm to 110 mm) in the setpoint.

Controllers	Rise time (s)	Overshoot (mm)	ISE (for 300 s)
PID	26	3.70	6580
DMC	18	4.50	5725
WDMC	24	1.90	5140

Table 6.14 Comparison of the performances of different controllers in the liquid level process for a pulse change of –20 mm (from 80 mm to 60 mm) in the level.

Controllers	Rise time (s)	Overshoot (mm)	ISE (for 300 s)
PID	25	9.98	4811
DMC	25	3.92	4259
WDMC	20	3.83	3459
ANNC (Dasgupta, 1999, 2004)	26	3.33	4165

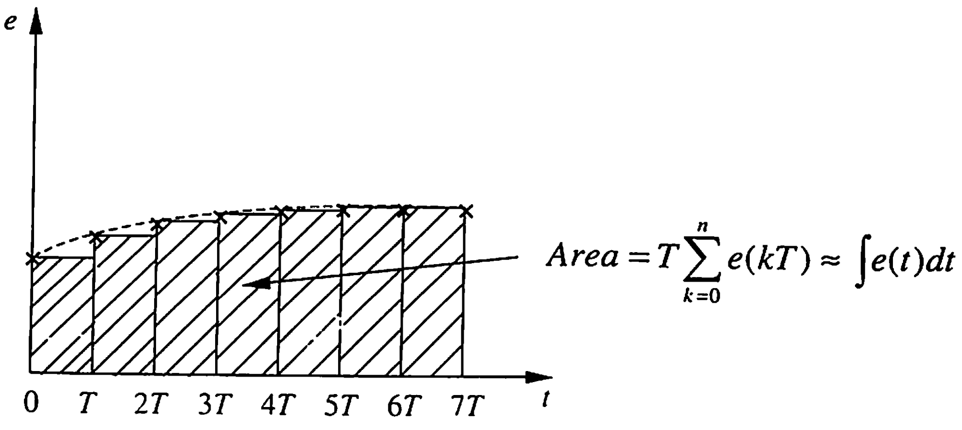


Figure 6.1 Numerical evaluation of integral.

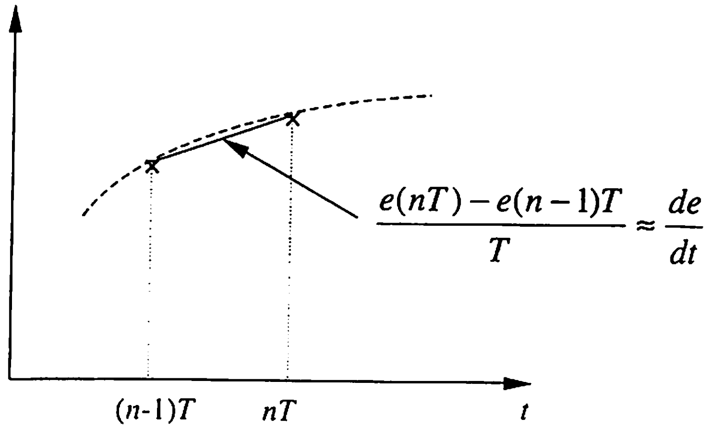


Figure 6.2 Numerical evaluation of derivative.

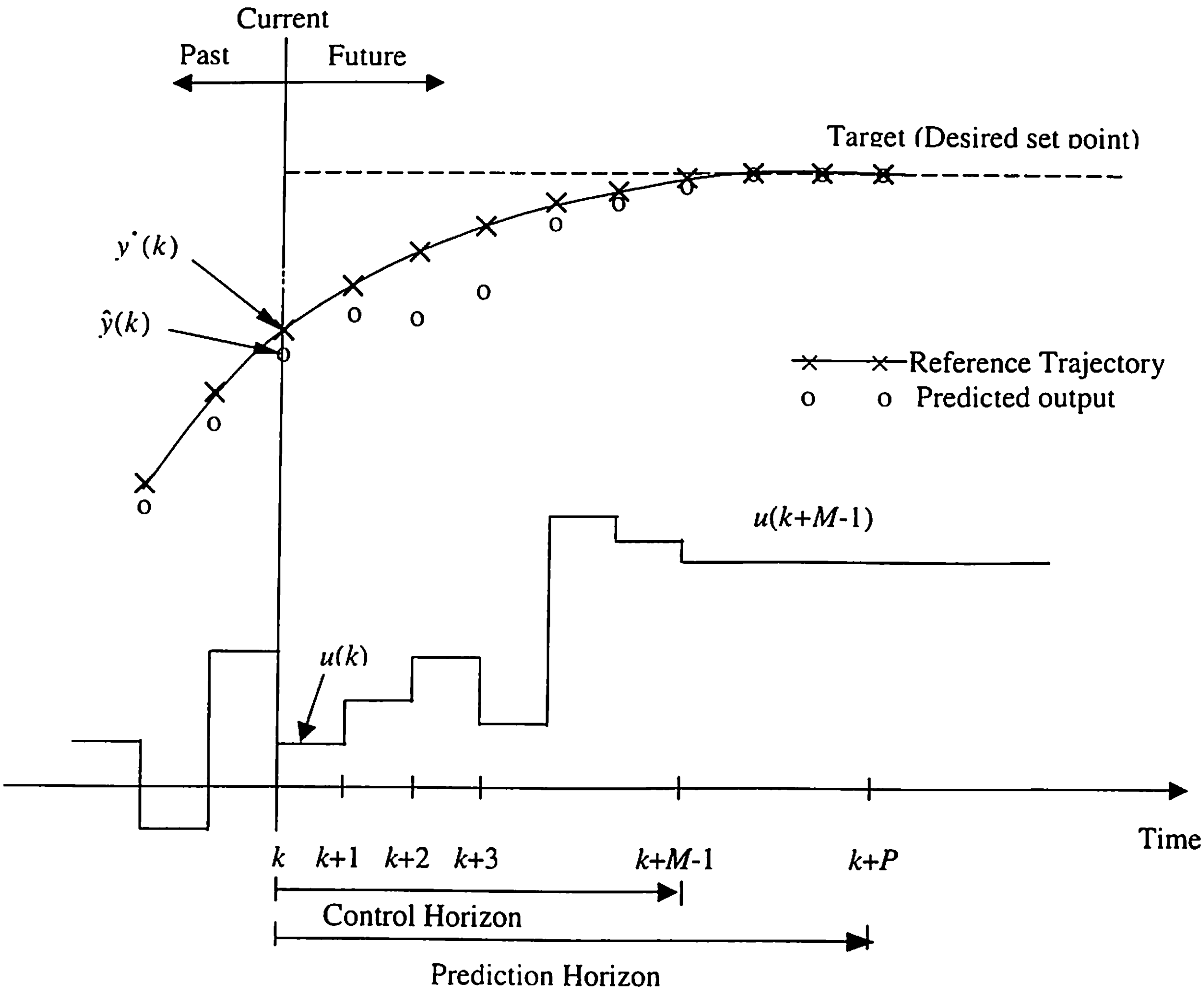


Figure 6.3 The basic concepts of DMC.

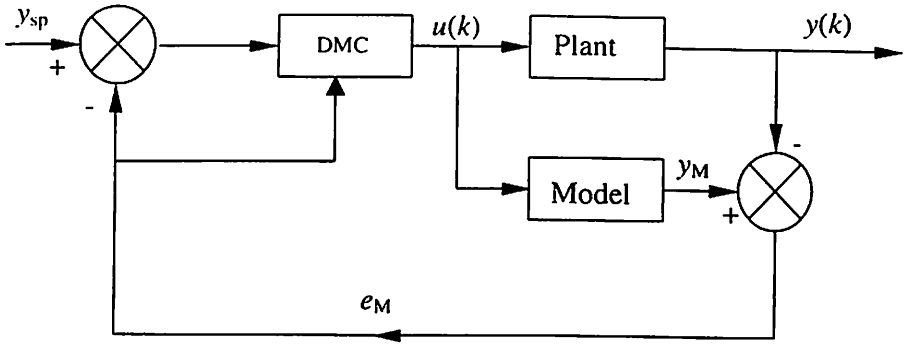


Figure 6.4 Block diagram of DMC.

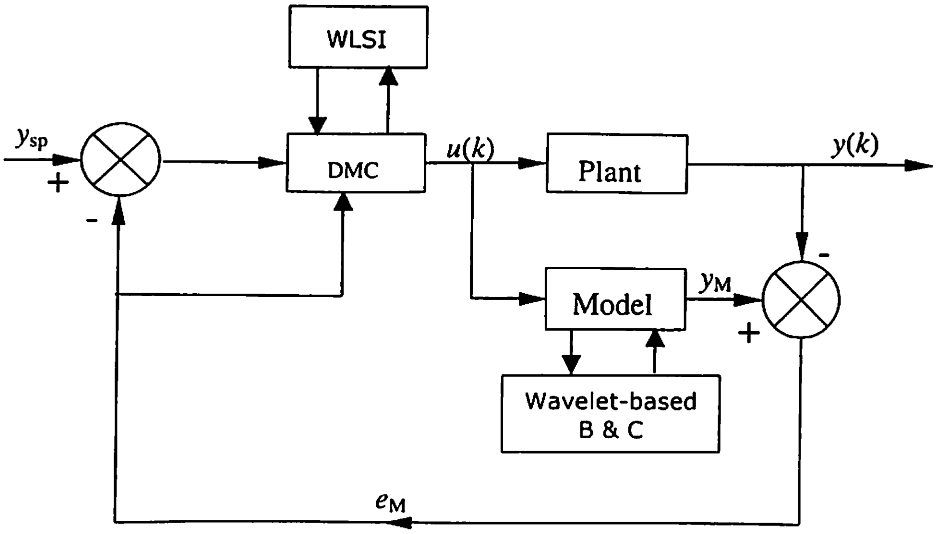


Figure 6.5 Block diagram of WDMC.

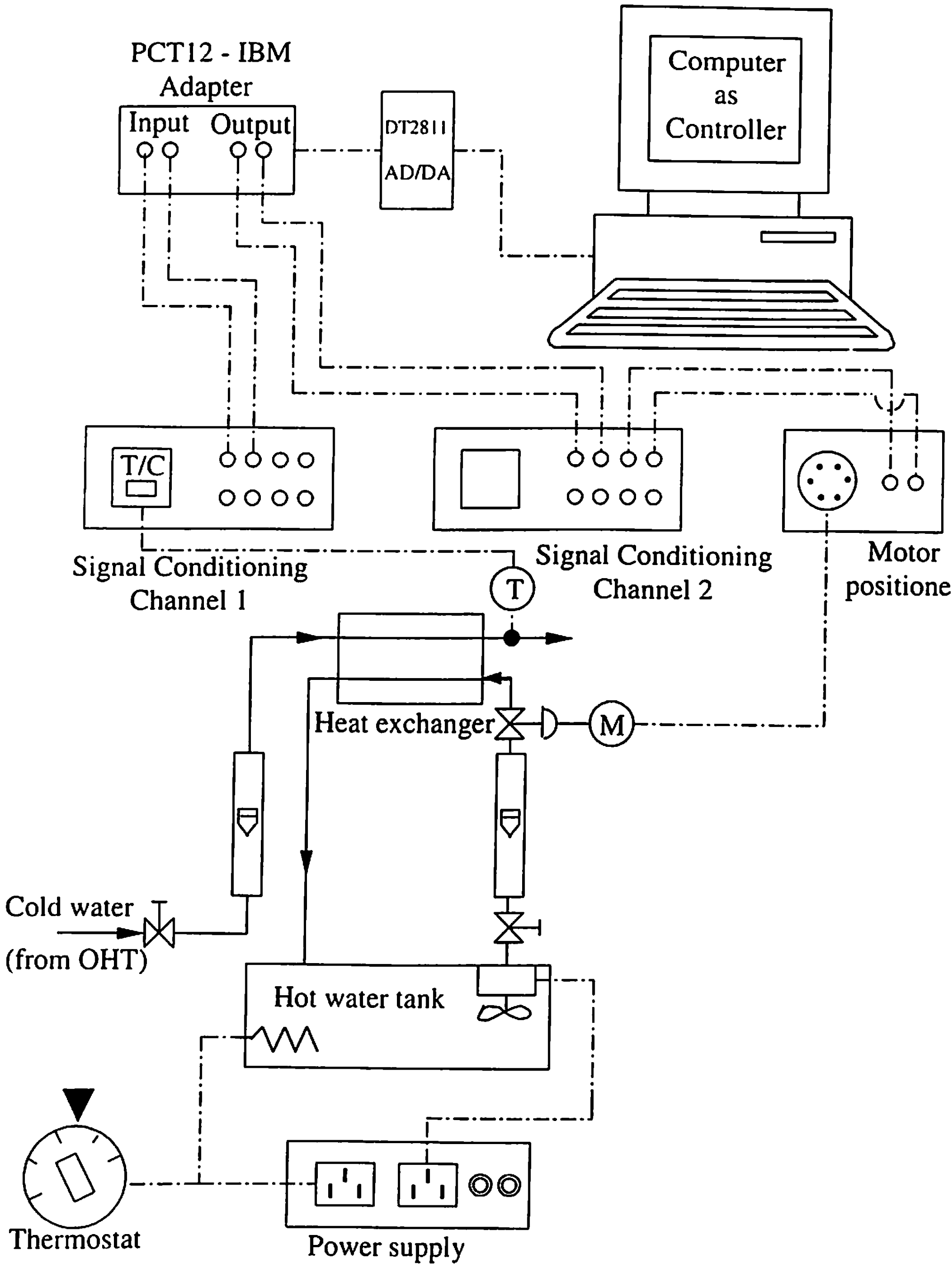


Figure 6.6 Circuit diagram for control of heat exchanger using computer.

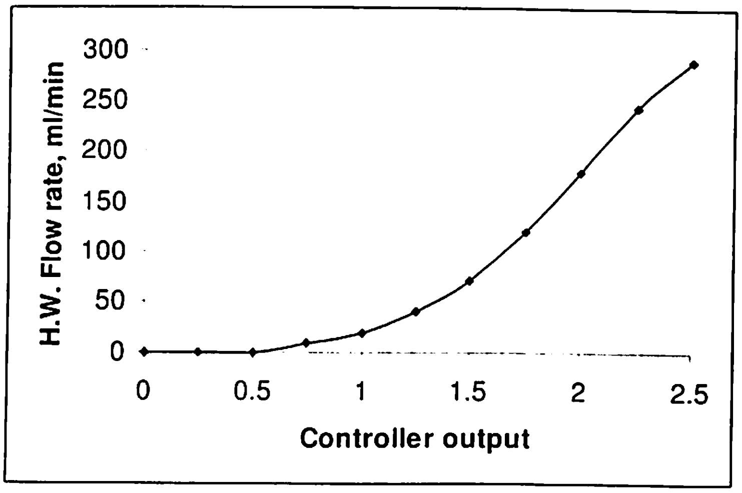


Figure 6.7 Characterization of the Motorized Valve.

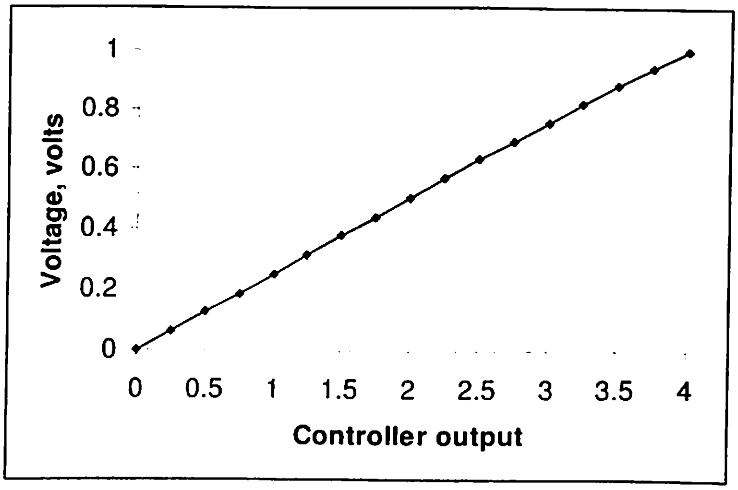


Figure 6.8 Relationships between Controller Output and Voltage.

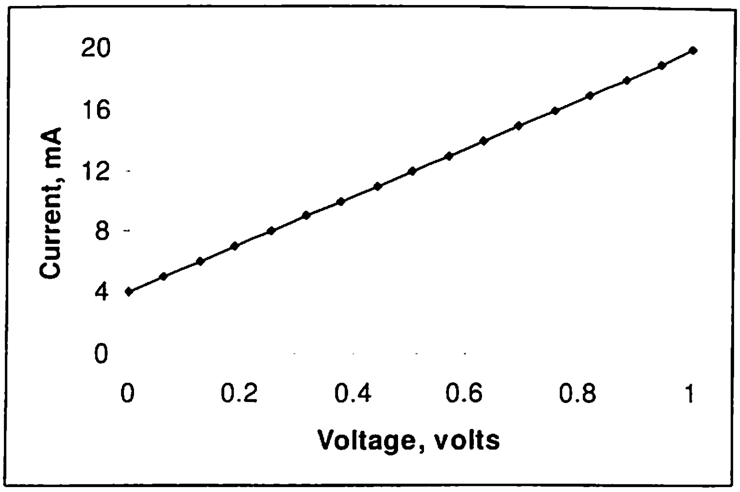


Figure 6.9 Conversion of voltage signal to current signal.

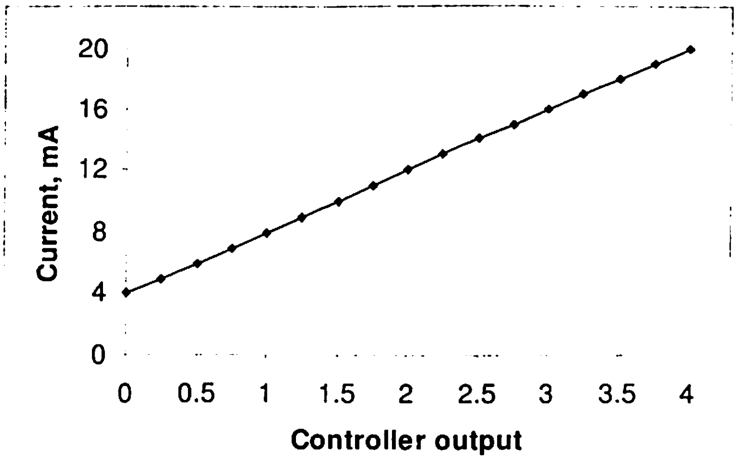


Figure 6.10 Relationship between controller output and current.

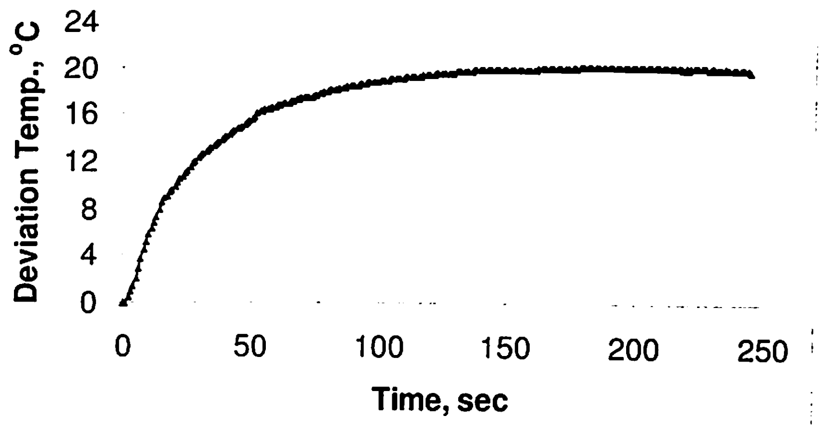


Figure 6.11 Process reaction curve of the heat exchanger process for a step change of 150 cc/min in the hot water flow rate keeping cold water flow rate constant at 200 cc/min (Servo Problem).

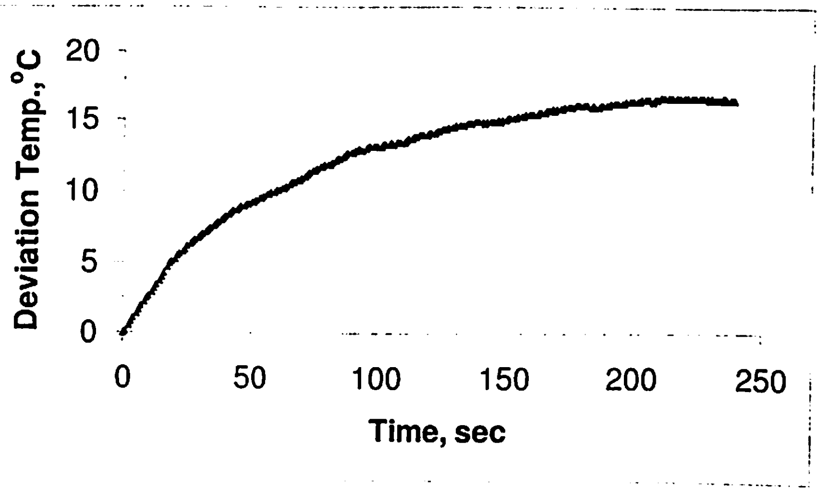


Figure 6.12 Process reaction curve of the heat exchanger process for a step change of –130 cc/min in the cold water flow rate, keeping hot water flow rate constant at 85 cc/min (regulator problem).

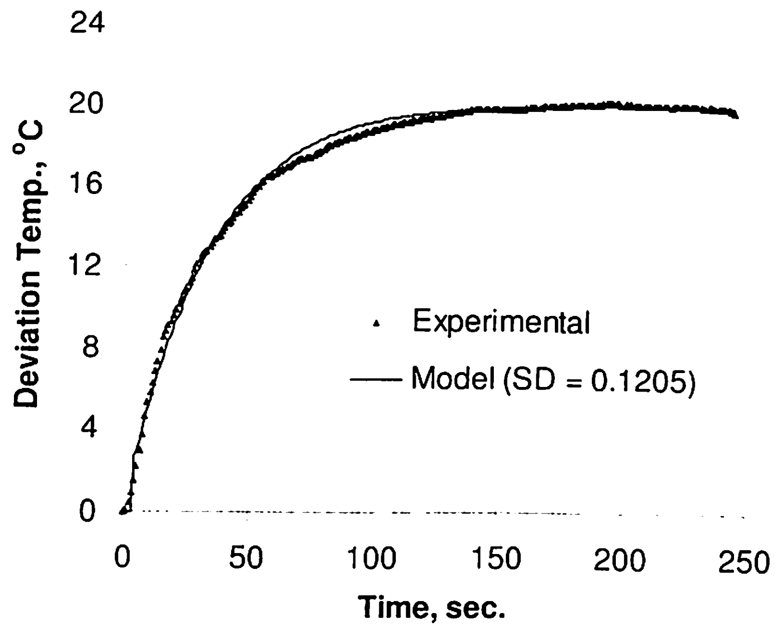


Figure 6.13 Validation of the FOPDT model for the heat exchanger process for servo problem.

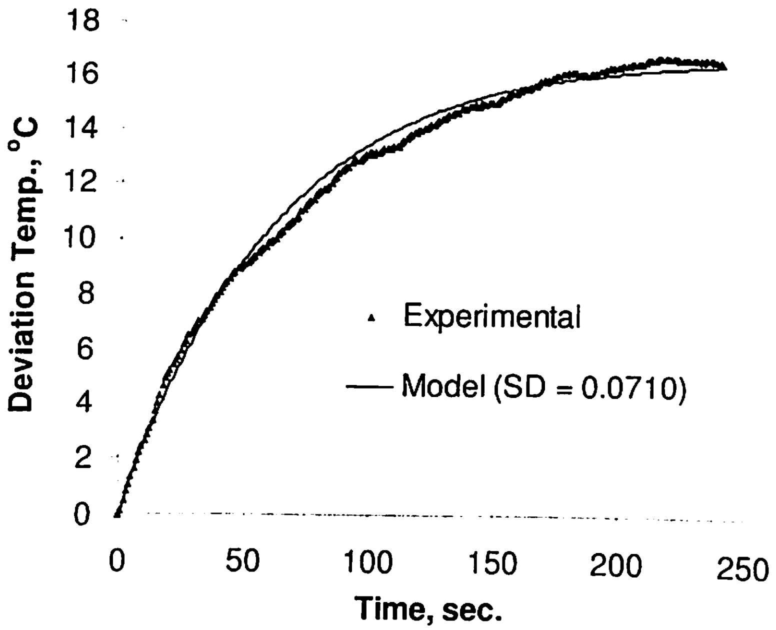


Figure 6.14 Validation of the FOPDT model for the heat exchanger process for regulatory problem.

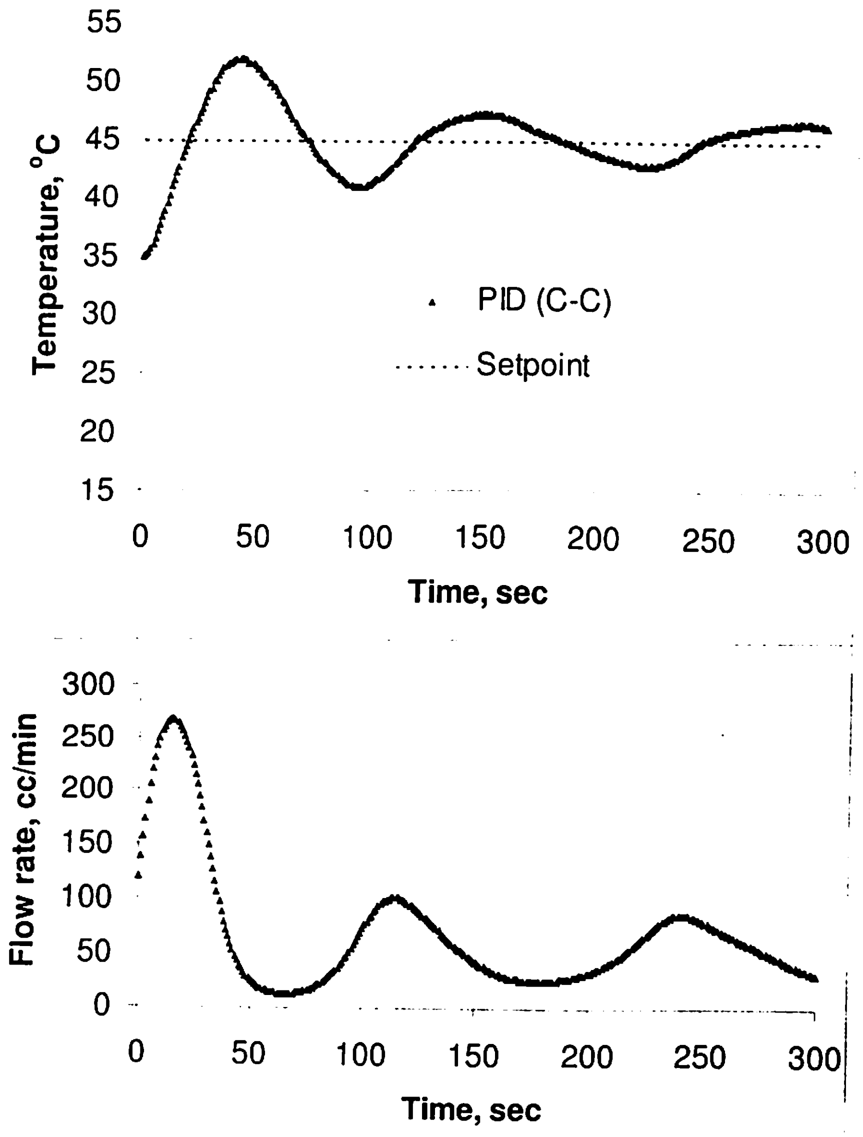


Figure 6.15 Closed loop response of the heat exchanger process using Cohen-Coon Settings for the PID controller (top) and corresponding hot water flow rate (bottom).

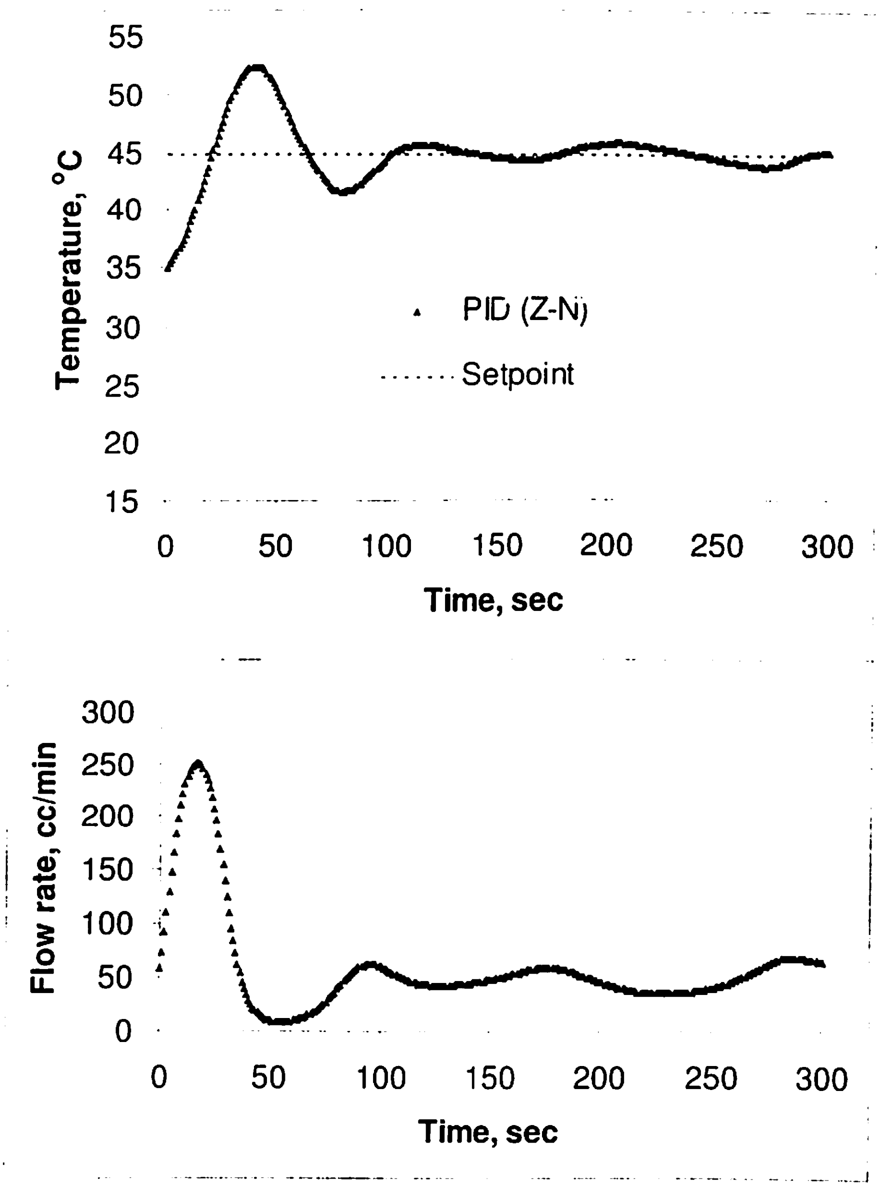


Figure 6.16 Closed loop response of the heat exchanger process using Ziegler-Nichols settings for the PID controller (top) and corresponding hot water flow rate (bottom).

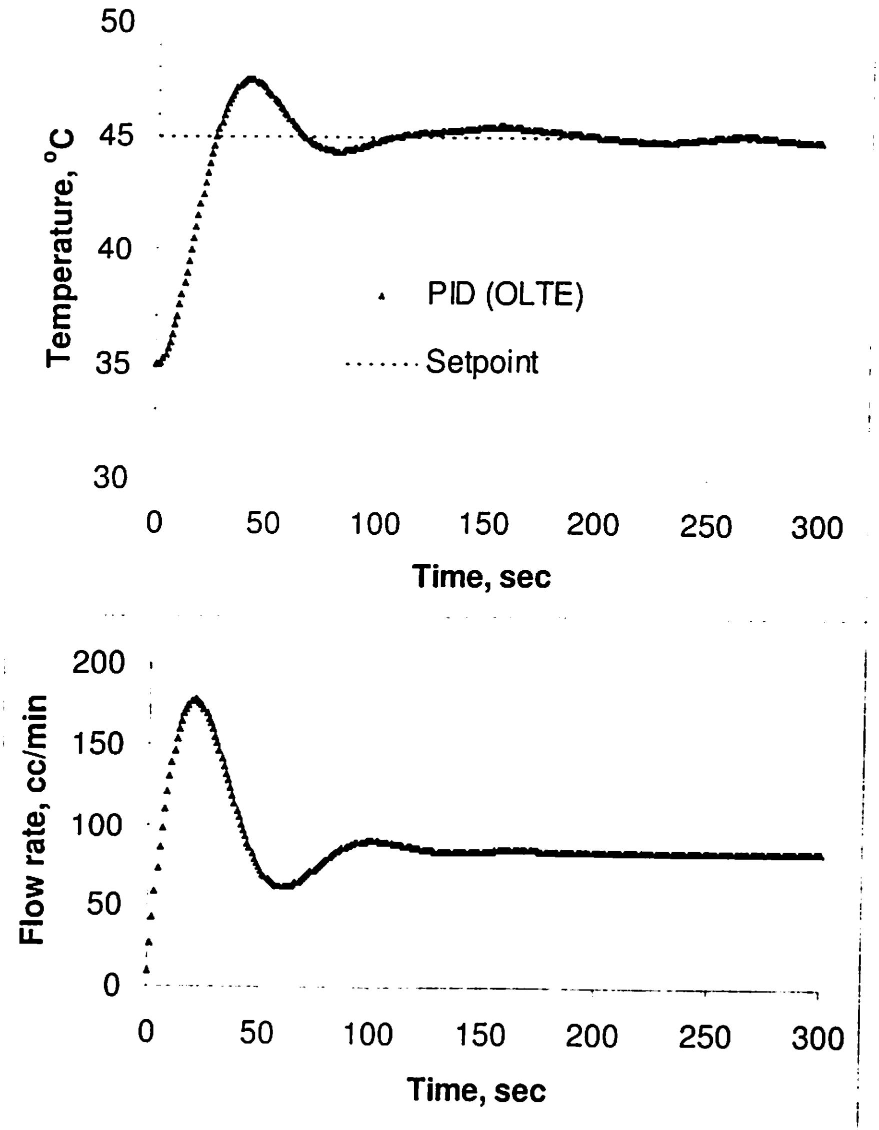


Figure 6.17 Closed loop response of the heat exchanger process for a step change of 10°C in the set point (35°C to 45°C) using on-line trial and error (OLTE) settings for the PID controller (top) and corresponding hot water flow rate (bottom).

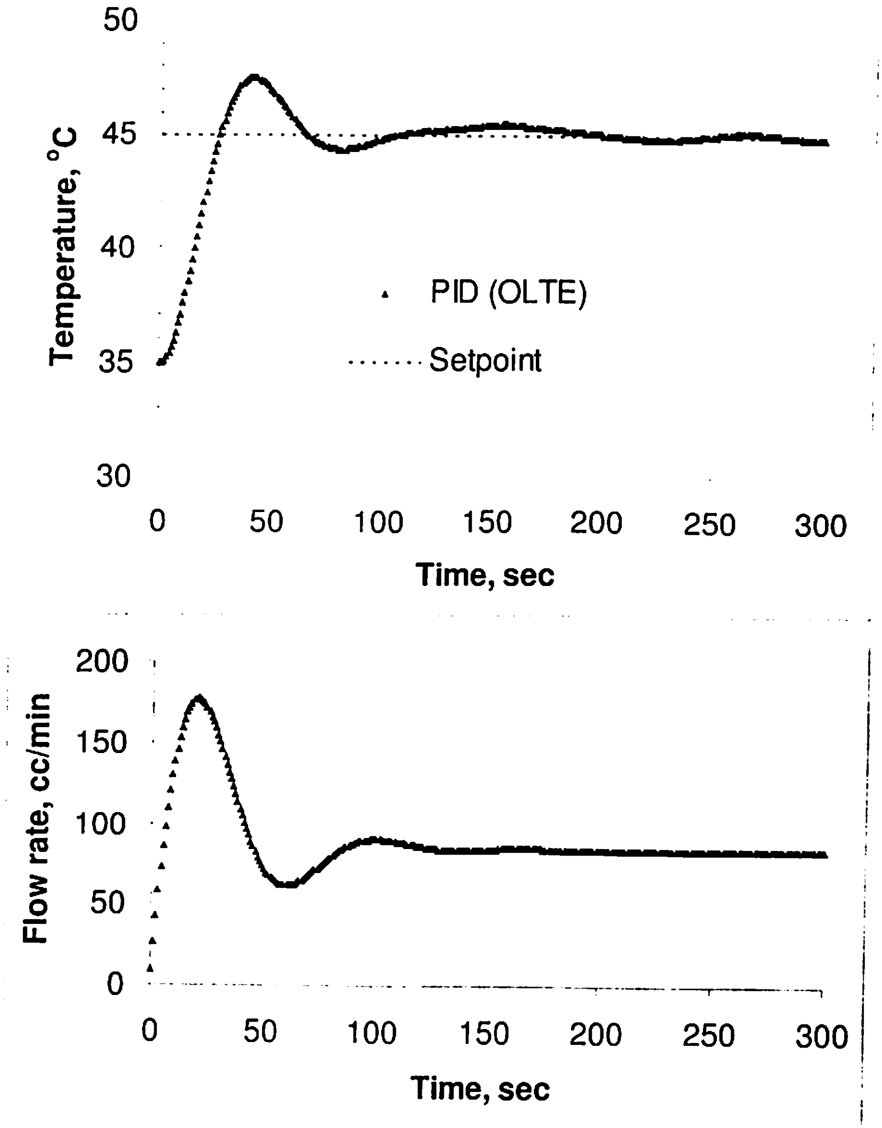


Figure 6.17 Closed loop response of the heat exchanger process for a step change of 10°C in the set point (35°C to 45°C) using on-line trial and error (OLTE) settings for the PID controller (top) and corresponding hot water flow rate (bottom).

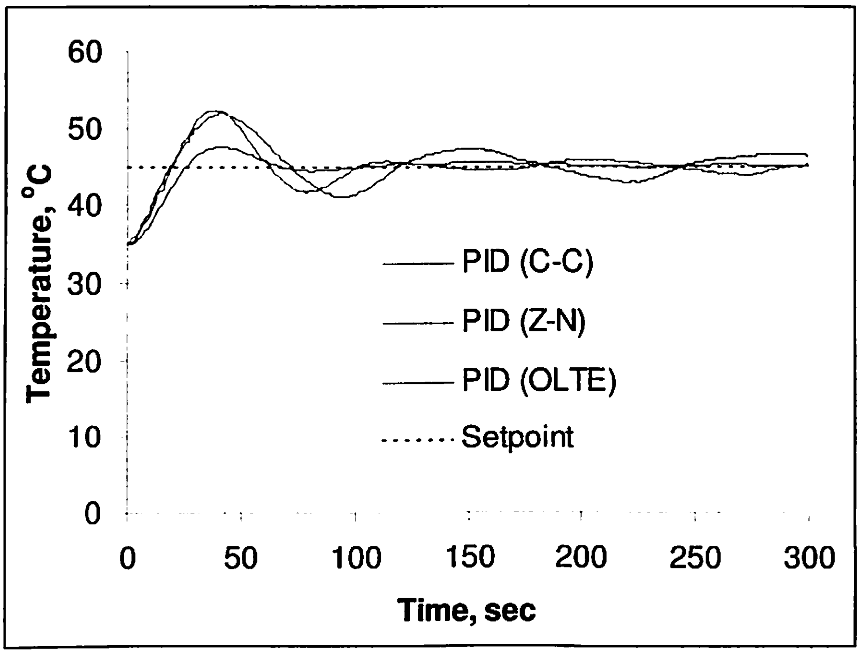


Figure 6.18 Comparison of the performances of PID controllers with different settings in the heat exchanger process for a step change from 35°C to 45°C in the cold water outlet temperature (Servo problem).

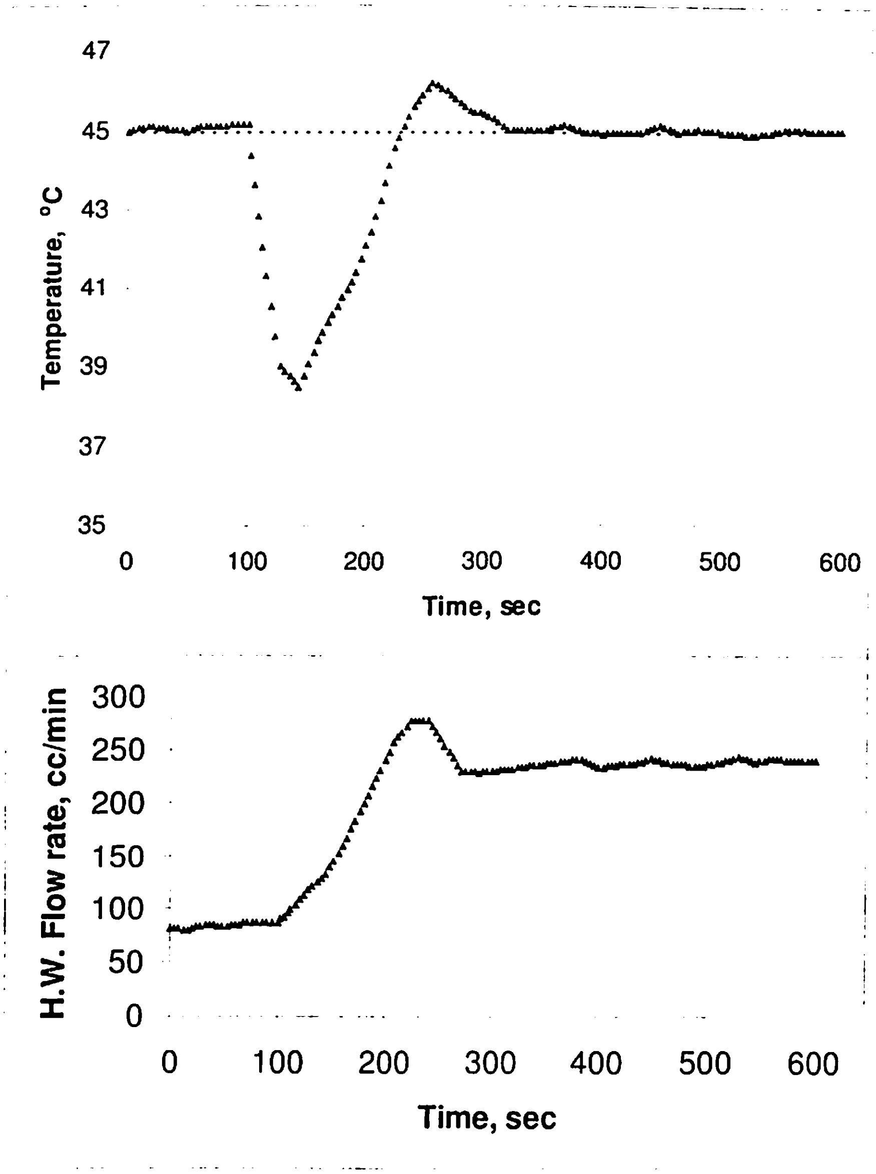


Figure 6.19 Closed-loop response of the heat exchanger process for a regulatory PID control, when a step disturbance of 100 cc/min is given in the cold water flow rate (top) and the corresponding hot water flow rate (bottom).

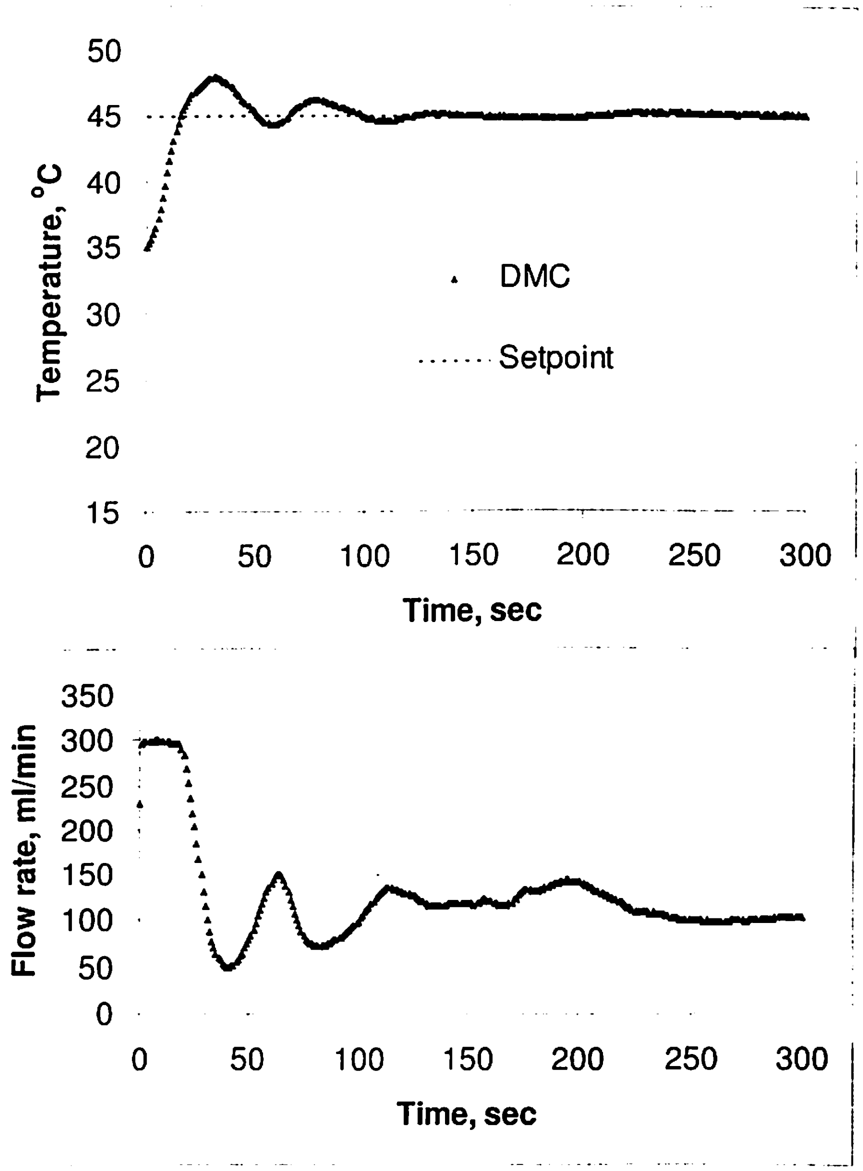


Figure 6.20 Closed-loop response of the heat exchanger process using DMC with NP=32, NC=16, NT=17 and MSP=200 (top) and corresponding hot water flow rate (bottom).

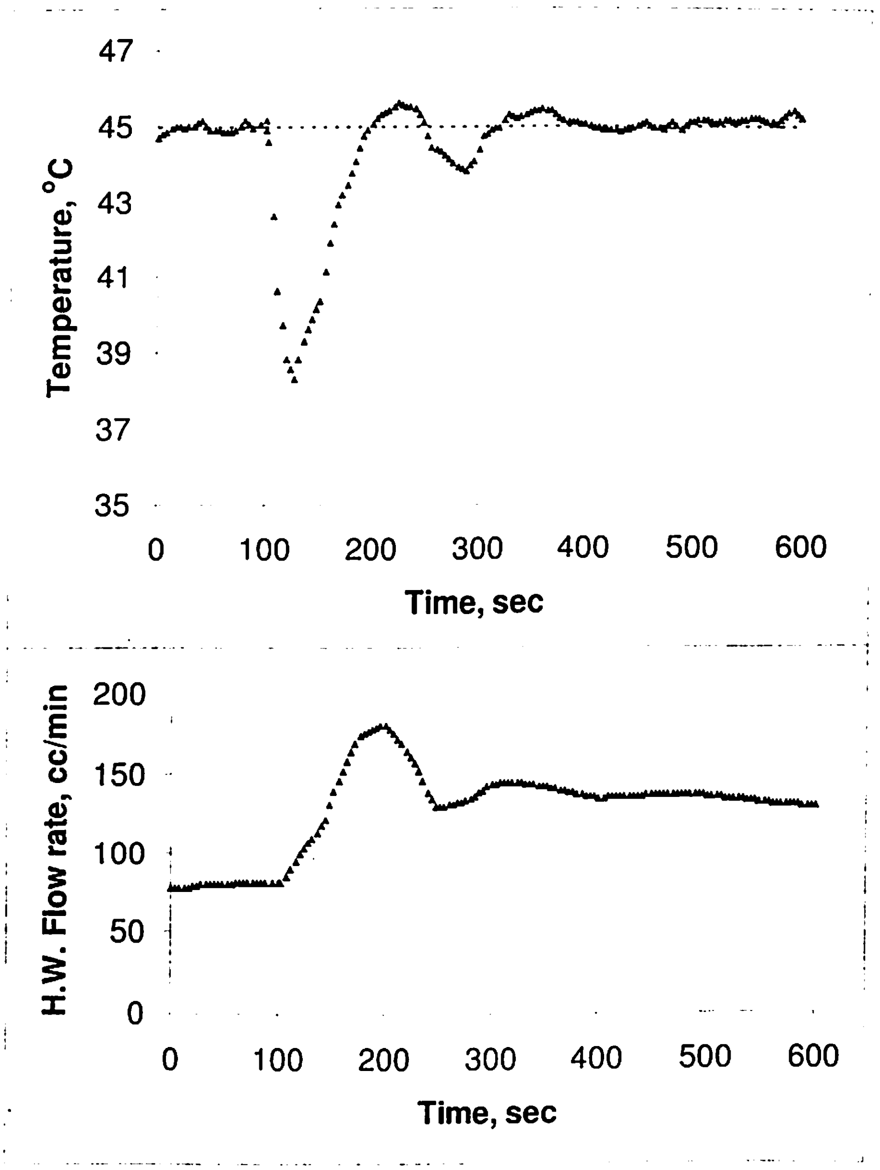


Figure 6.21 Closed-loop response of the heat exchanger process for a regulatory DMC control, when a step disturbance of 100 cc/min is given in the cold water flow rate (top) and the corresponding hot water flow rate (bottom).

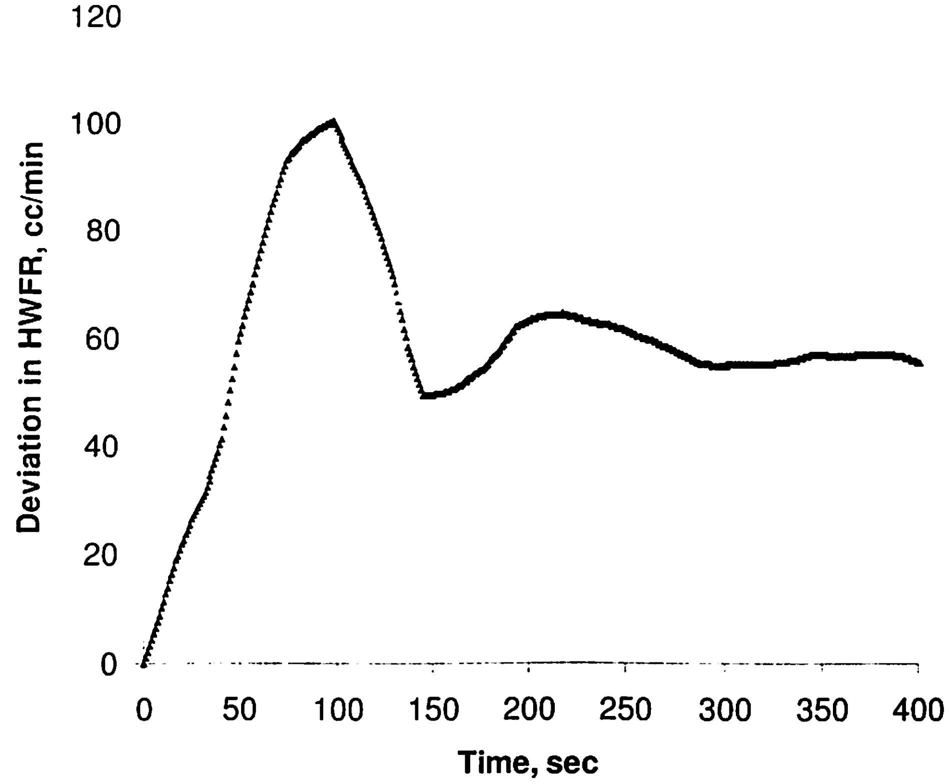


Figure 6.22 Blocking coefficients (values of hot water flow rate in deviation form at each second interval) for Wavelet-based controller obtained from a regulatory DMC of the heat exchanger process for a step change in the cold water flow rate.

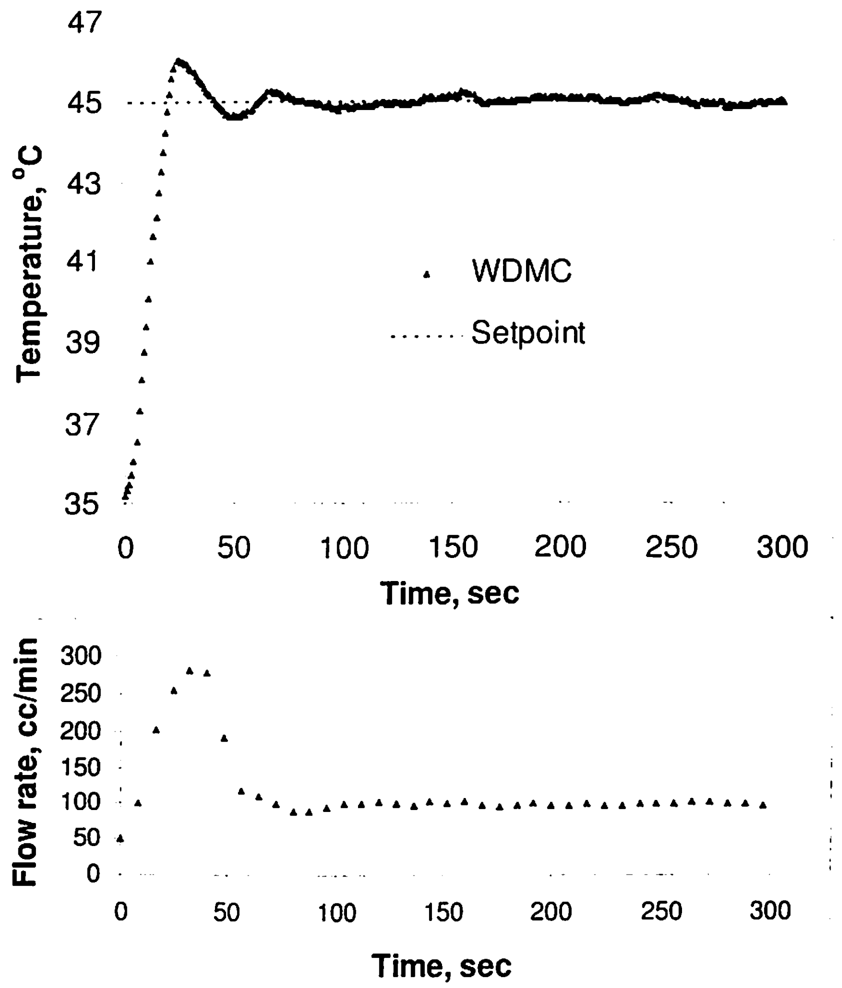
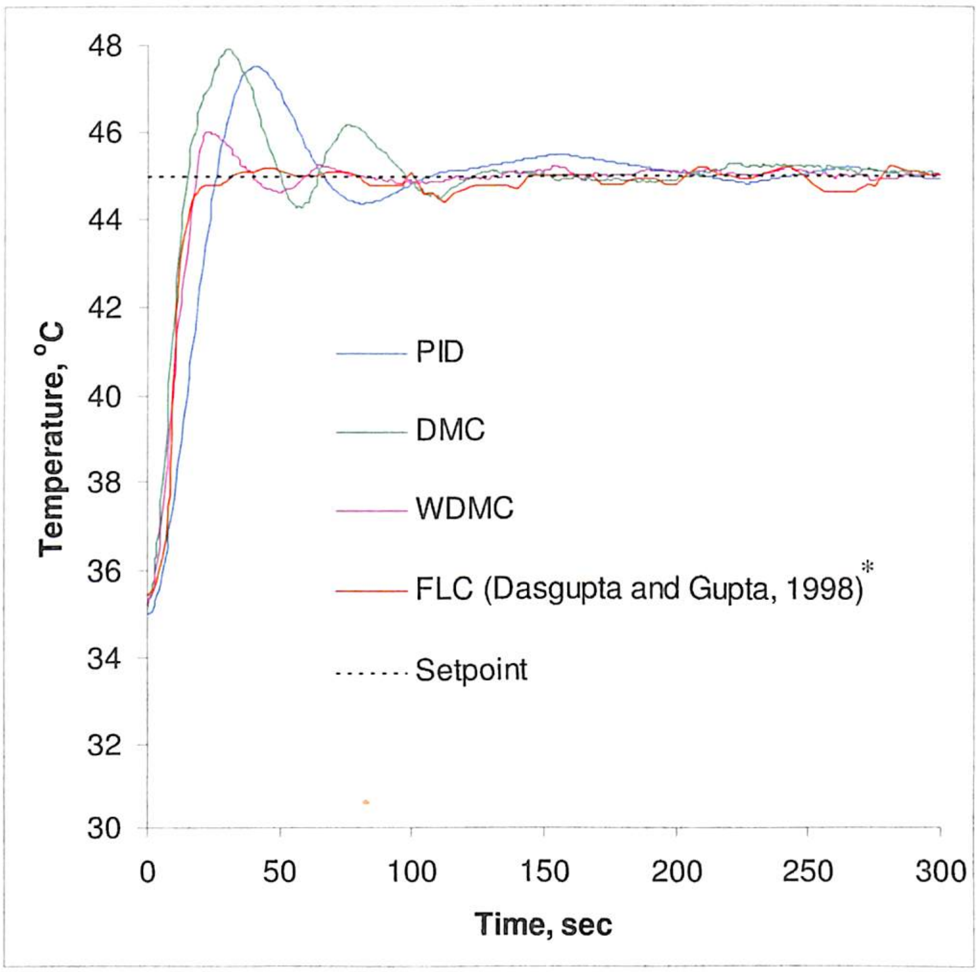


Figure 6.23 Closed loop response of the heat exchanger process using WDMC for a step change of 10°C in the set point (35°C to 45°C) with tuned parameters NP=32, NC=16, NT=17, MSP=200, d_y =0.02, d_u =0.05 (top) and corresponding hot water flow rate (bottom).



*The data for Fuzzy logic controller (FLC) were generated using X-Y grids from the work of Dasgupta and Gupta (1998) for comparison purpose only.

Figure 6.24 Comparison of the performances of different controllers in servo control problem for a step change from 35°C to 45°C in the setpoint of the cold water outlet temperature.

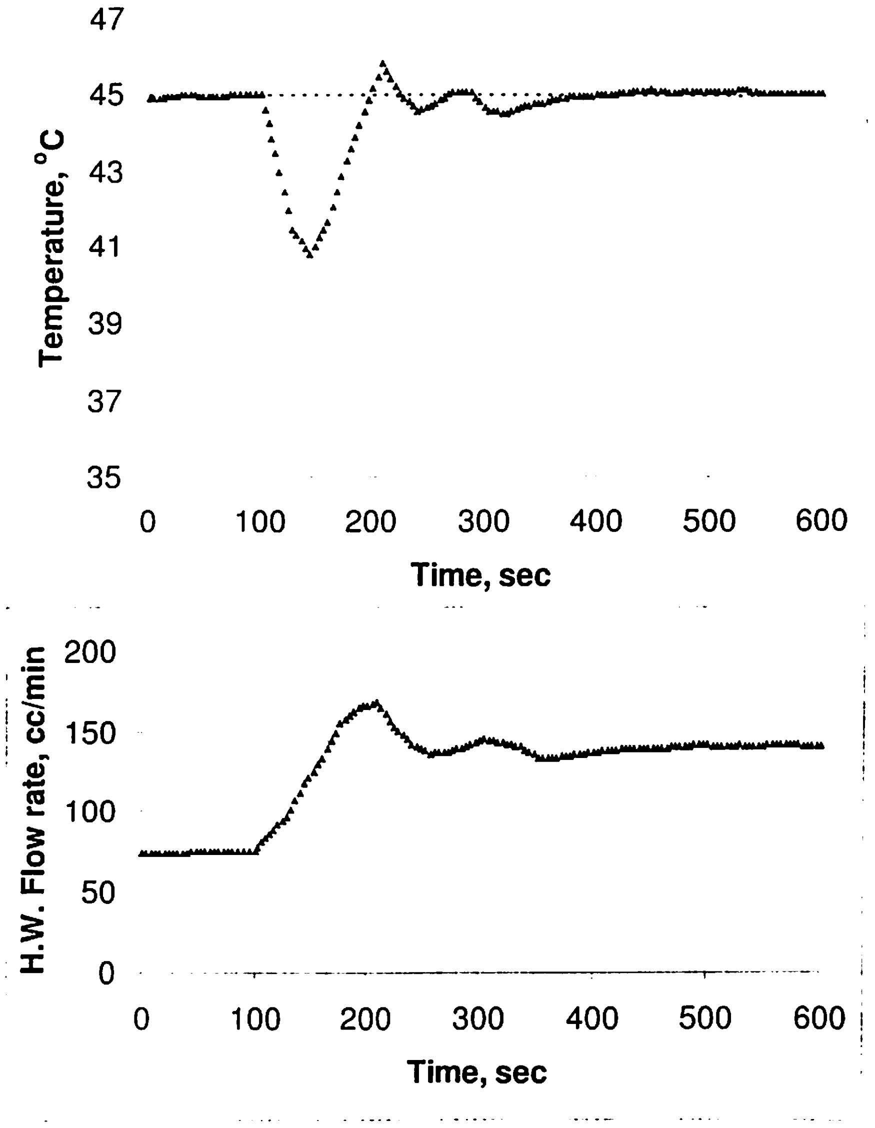


Figure 6.25 Closed loop response of the heat exchanger process using WDMC for a step change of 100 cc/min in the cold water flow rate.

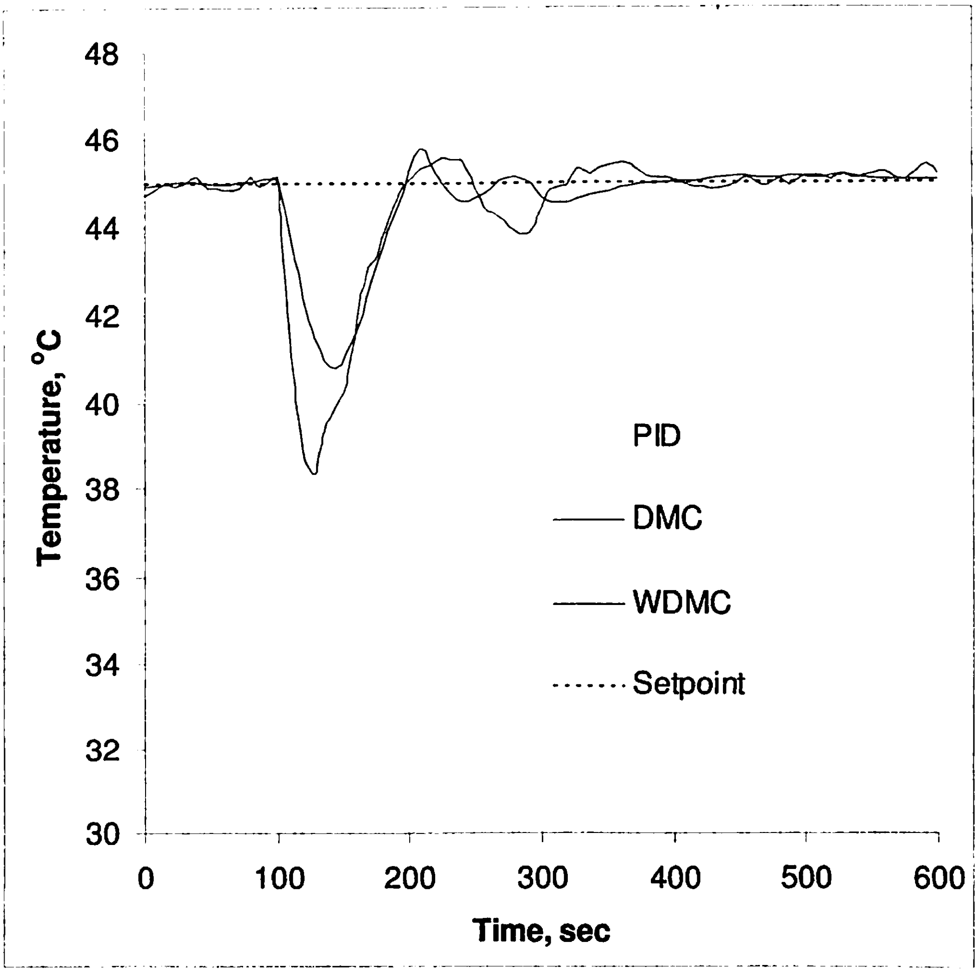


Figure 6.26 Comparison of the performances of different controllers in the heat exchanger process for a step change of 100 cc/min in the cold water flow rate (regulatory problem).

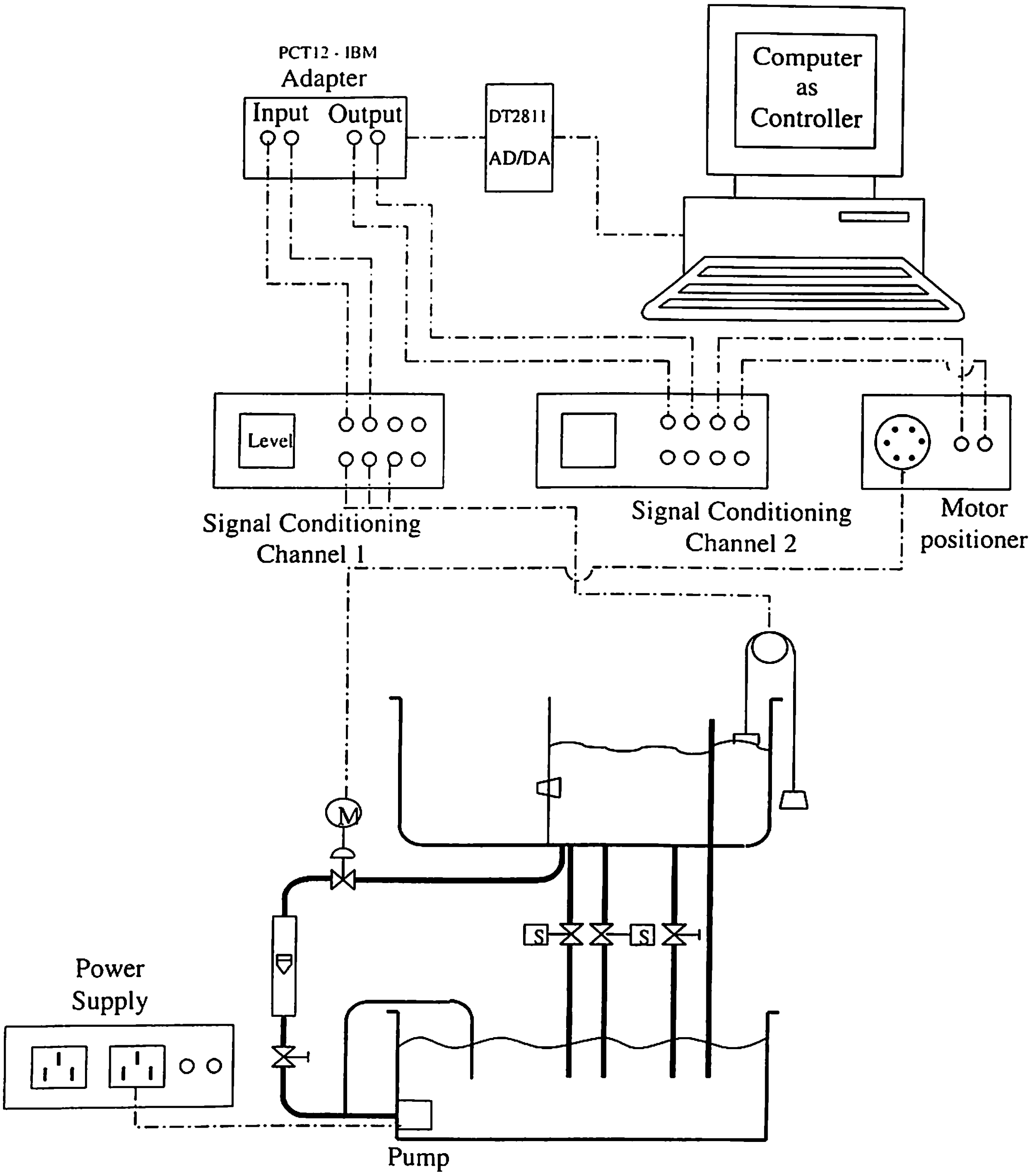


Figure 6.27 Circuit diagram for control of liquid level process using computer.

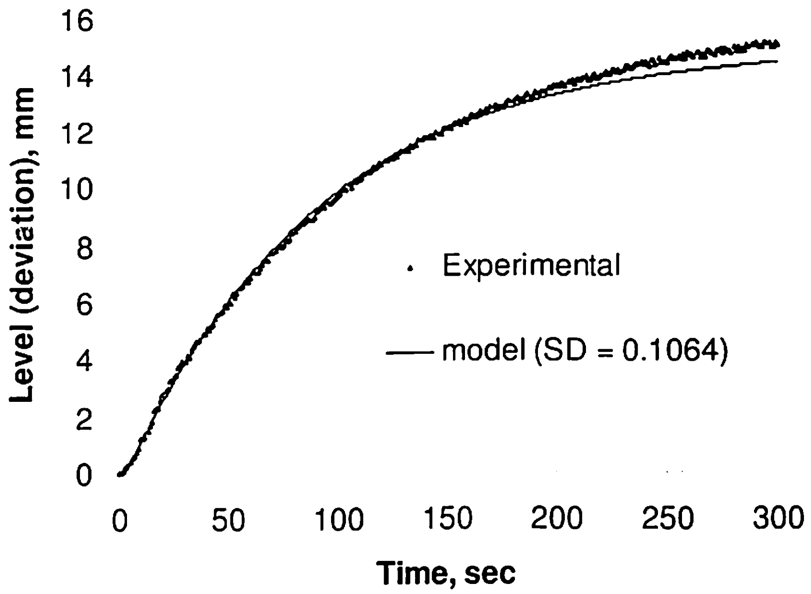


Figure 6.28 Open-loop response (process reaction curve) of the liquid level process for a step change of 50 cc/min in the inlet flow rate. Validation of the model obtained from Cohen-Coon method.

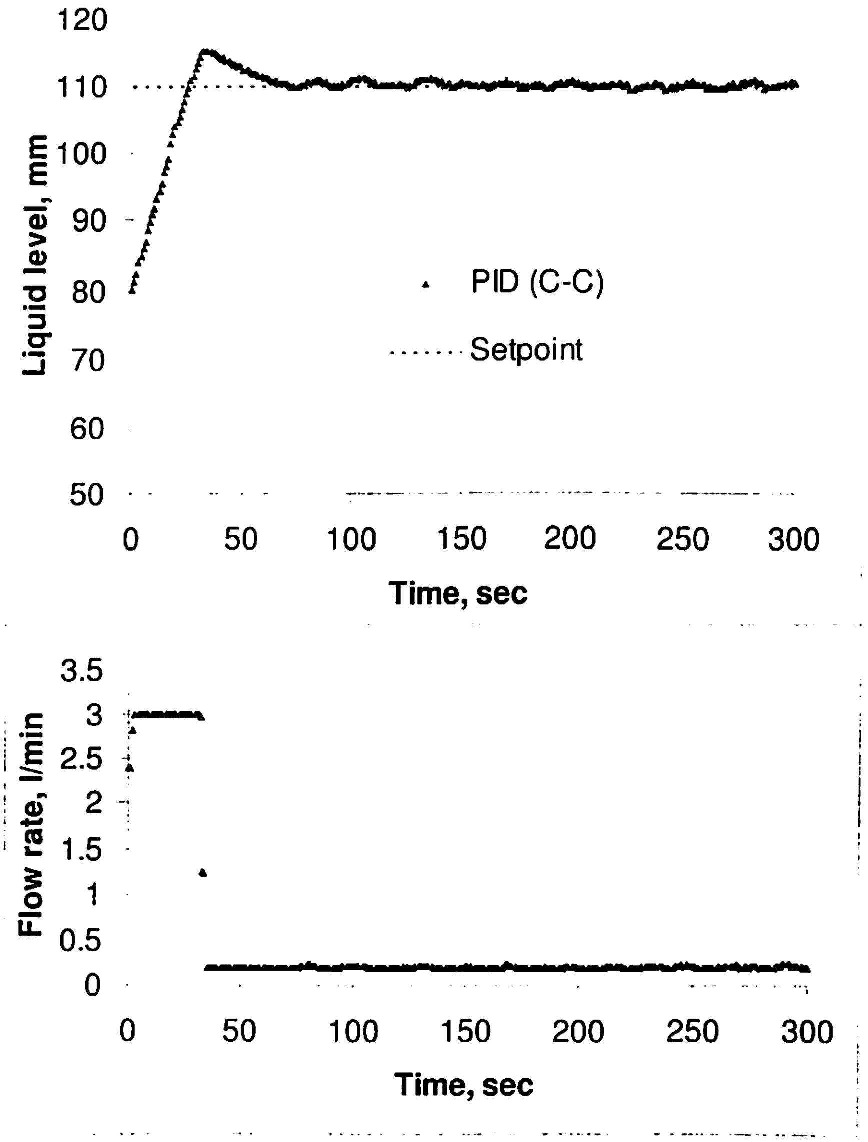


Figure 6.29 Closed loop response of the liquid level process using Cohen-Coon Settings of the PID controller (top) and corresponding inlet water flow rate (bottom) for a step change from 80 mm to 110 mm in the setpoint.

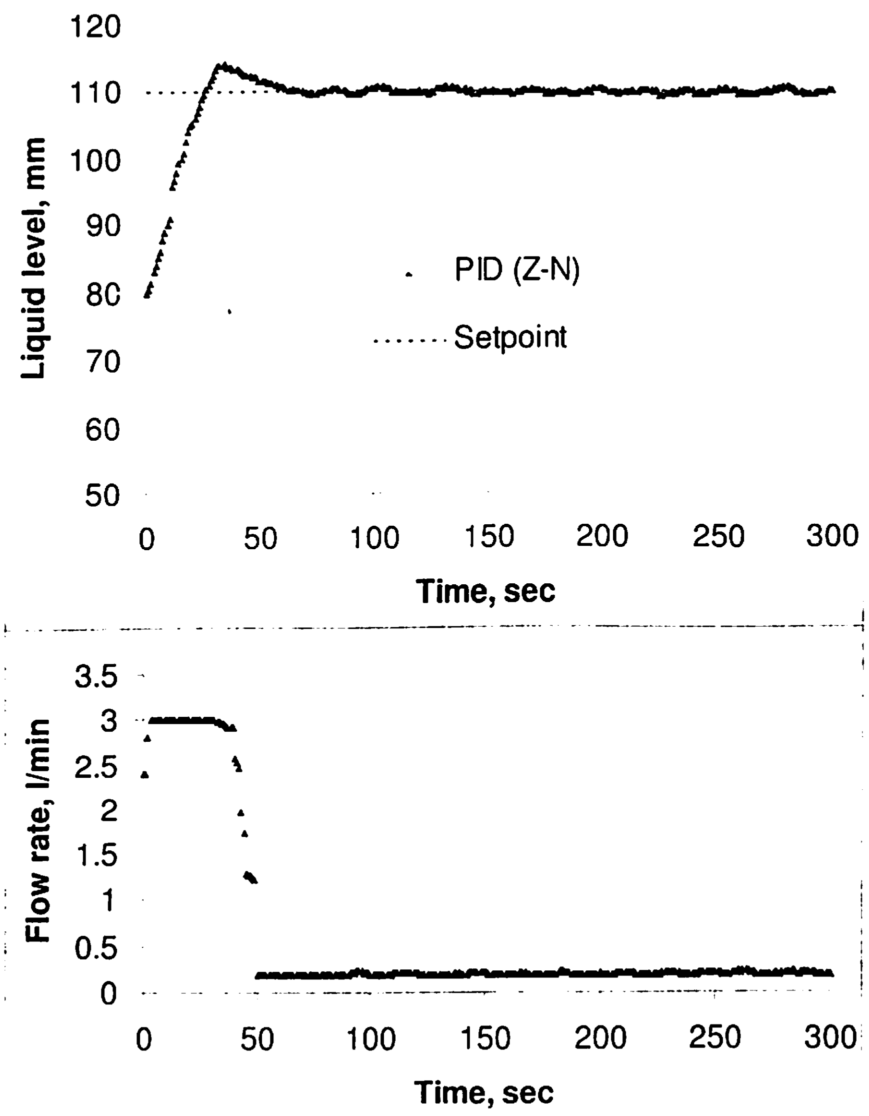


Figure 6.30 Closed loop response of the liquid level process using Ziegler-Nichols settings of the PID controller (top) and corresponding inlet water flow rate (bottom) for a step change from 80 mm to 110 mm in the setpoint.

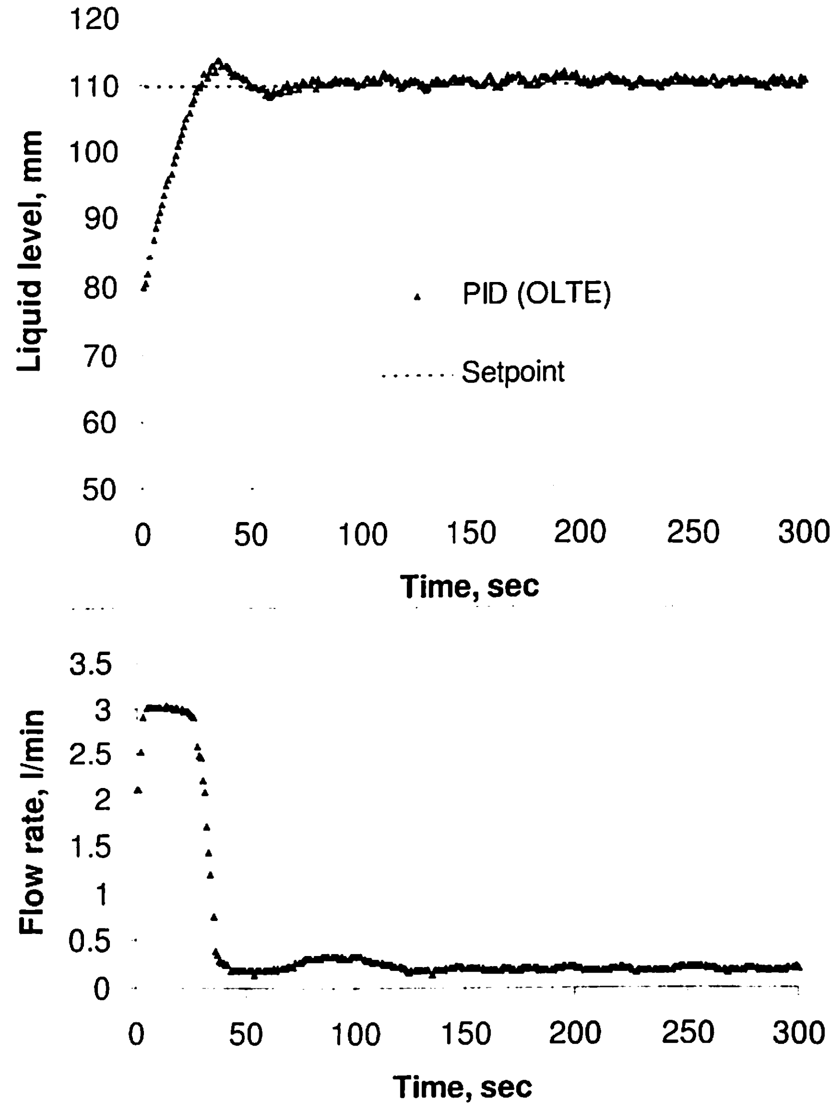


Figure 6.31 Closed loop response of the liquid level process for a step change from 80 mm to 110 mm in the setpoint using online-trial-and-error settings of the PID controller (top) and the corresponding inlet water flow rate (bottom).

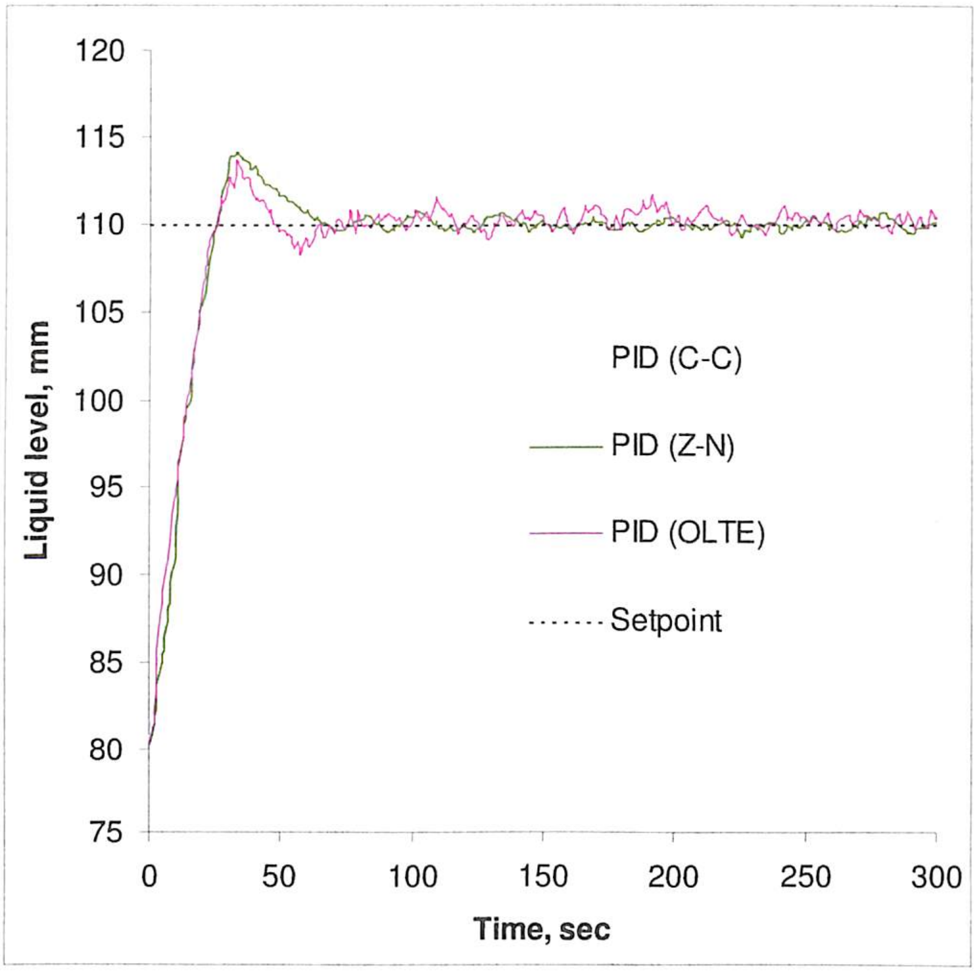


Figure 6.32 Comparison of the PID controllers with different settings for a step change of 30 mm in the set point for the liquid lever control process.

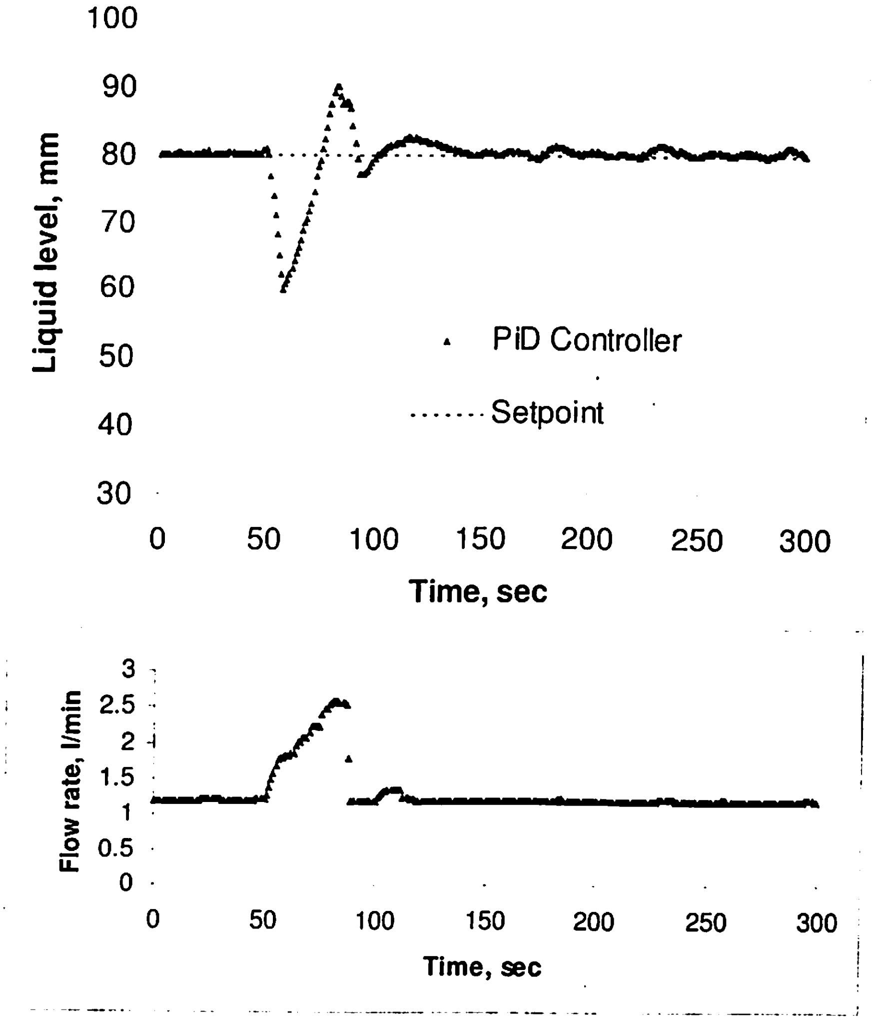


Figure 6.33 Closed loop response of the liquid level process using online-trial-and-error settings of the PID controller (top) and corresponding inlet water flow rate (bottom) for a pulse change from 80 mm to 60 mm in the level (regulatory problem).

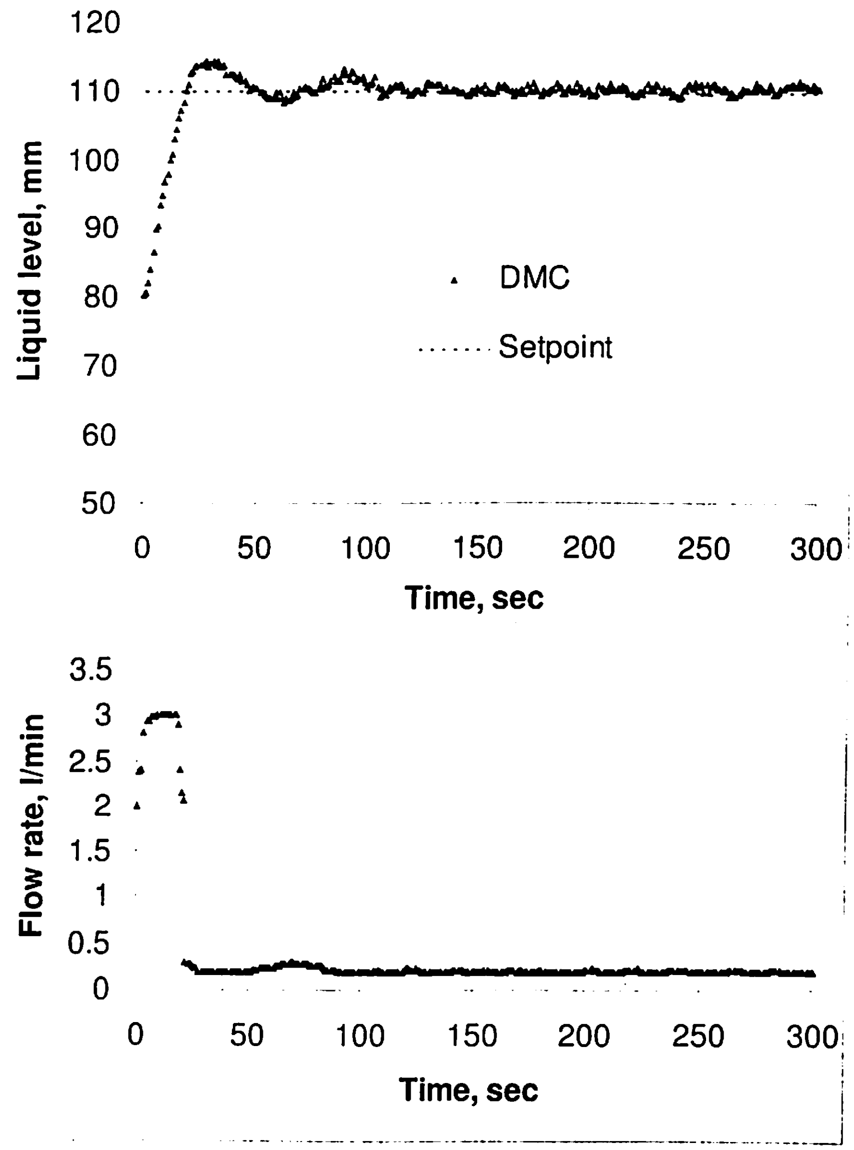


Figure 6.34 Closed loop response of the liquid level process using Dynamic Matrix Controller (top) and corresponding inlet water flow rate (bottom) for a step change from 80 mm to 110 mm in the setpoint.

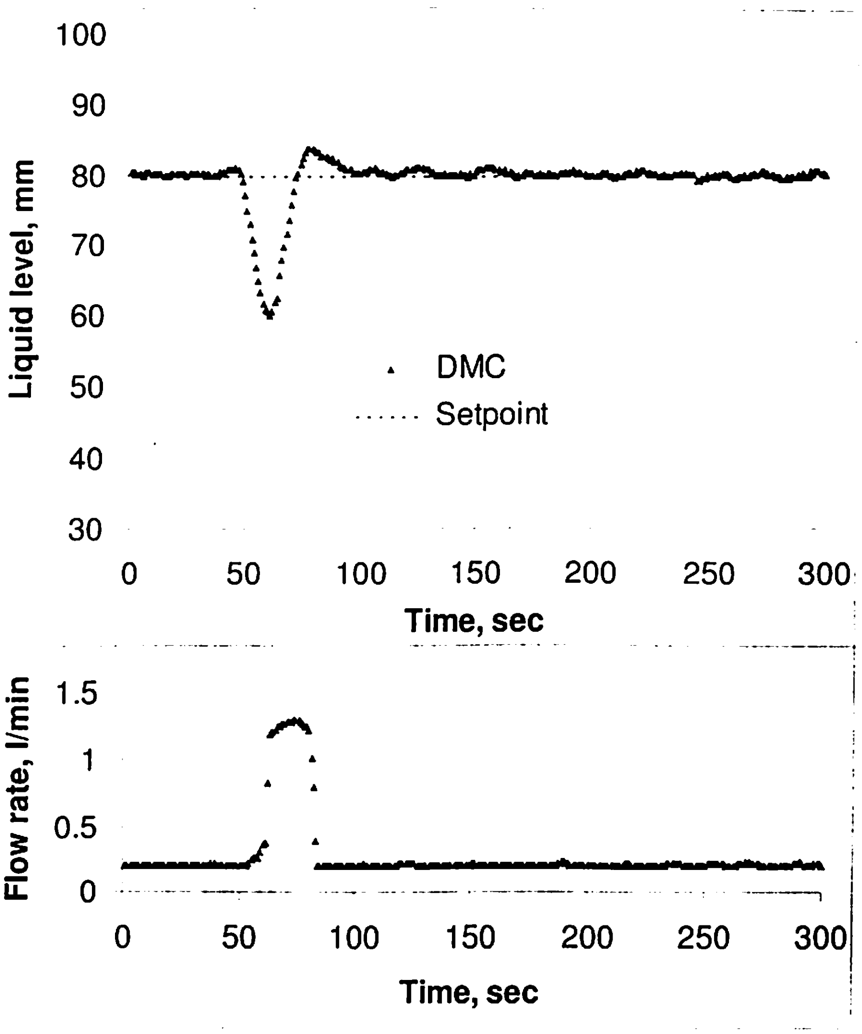


Figure 6.35 Closed loop response of the liquid level process using DMC (top) and corresponding inlet water flow rate (bottom) for a pulse change from 80 mm to 60 mm in the level (regulatory problem).

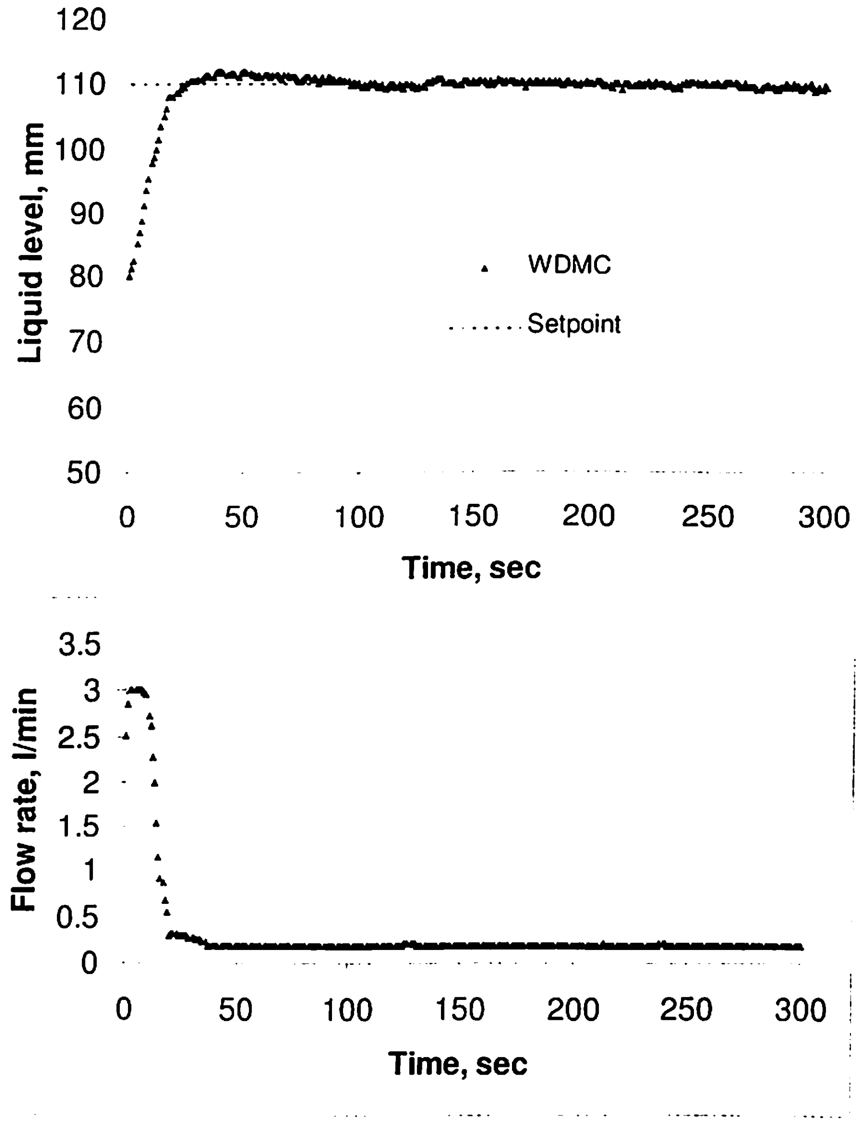


Figure 6.36 Closed loop response of the liquid level process using WDMC (top) and corresponding inlet water flow rate (bottom) for a step change from 80 mm to 110 mm in the setpoint.

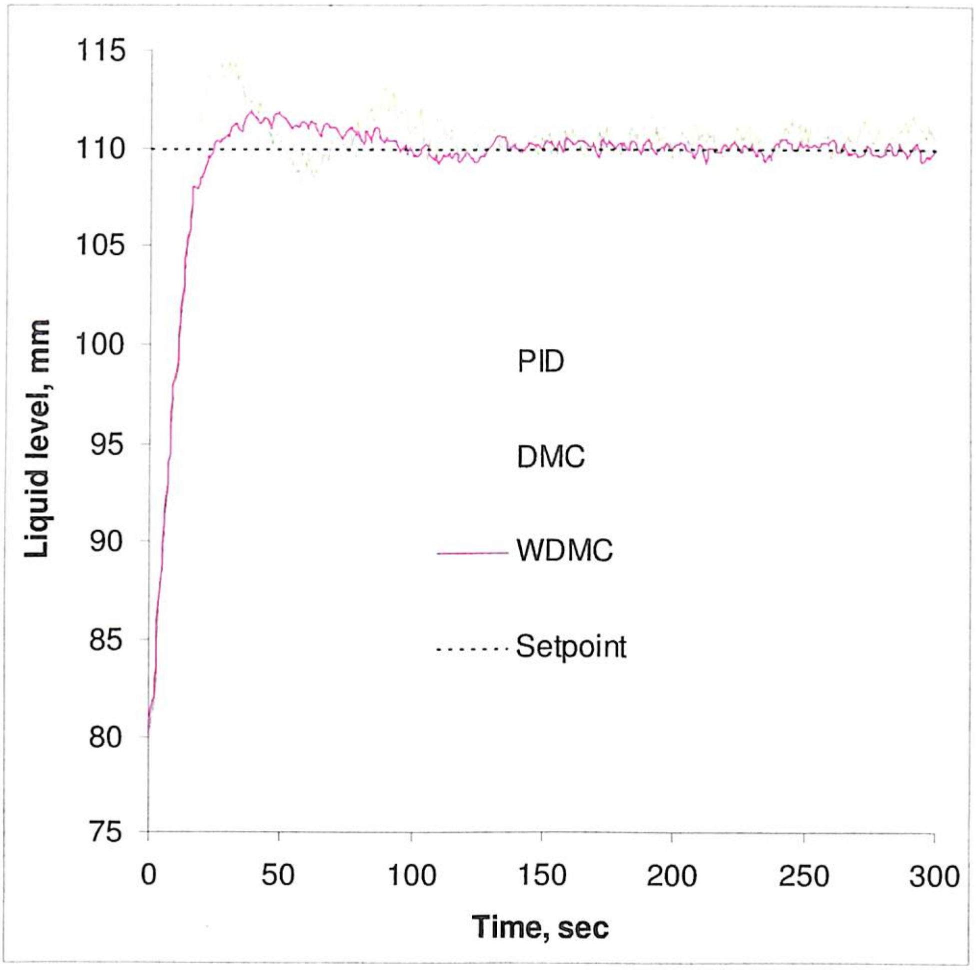


Figure 6.37 Comparison of the performances of different controllers for a step change from 80 mm to 110 mm in the setpoint for the liquid level process (servo problem).

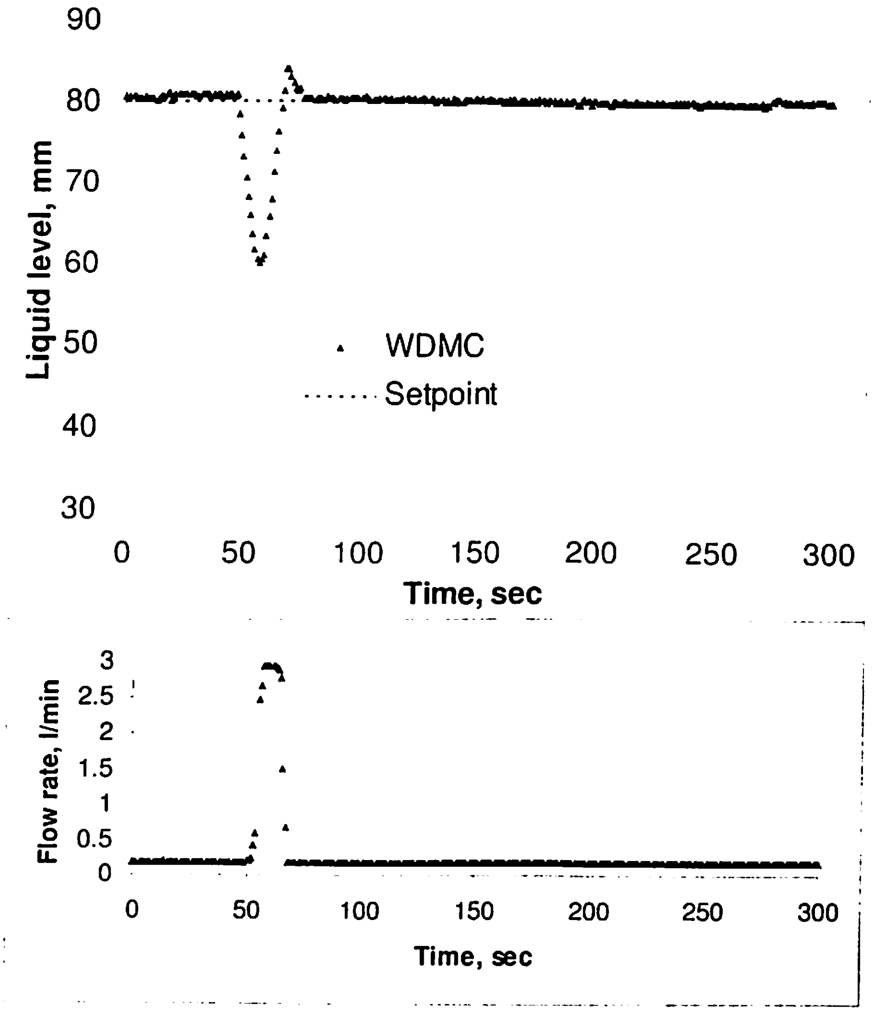
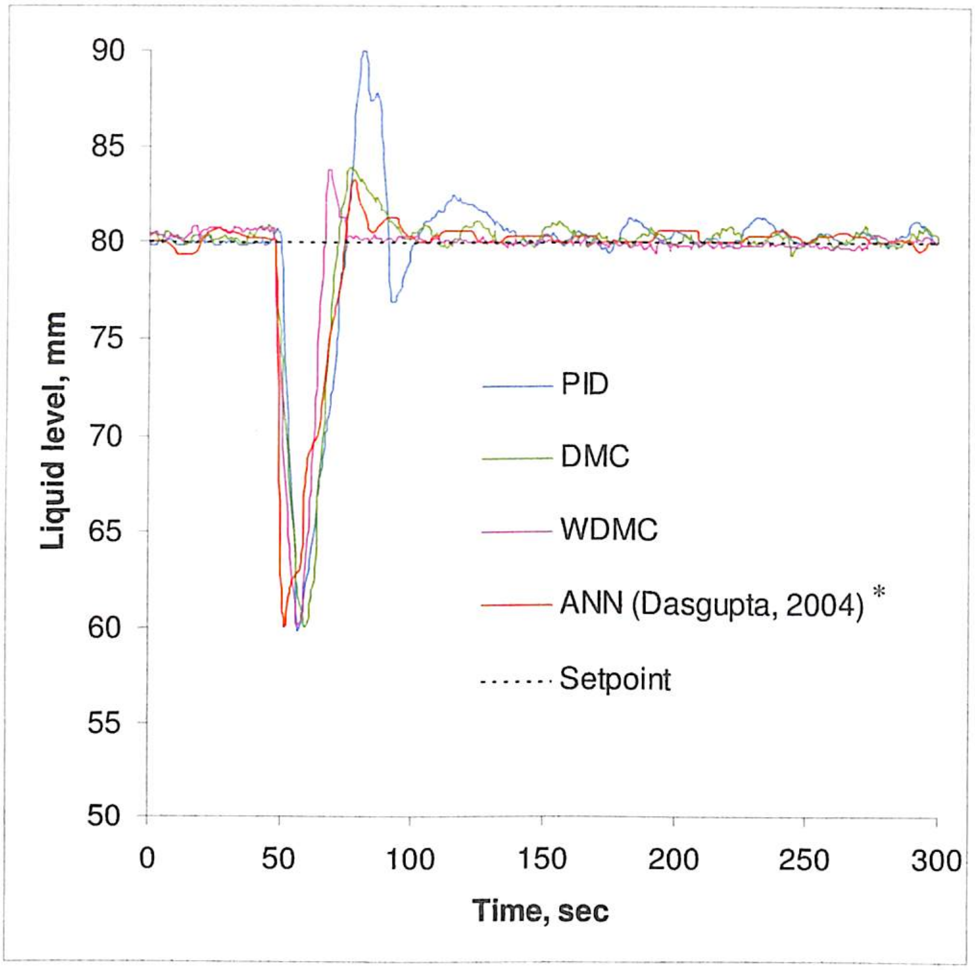


Figure 6.38 Closed loop response of the liquid level process using WDMC (top) and corresponding inlet water flow rate (bottom) for a pulse change from 80 mm to 60 mm in the level (regulatory problem).



*The data for ANN controller were generated using X-Y grids from the work of Dasgupta (2004) for comparison purpose only.

Figure 6.39 Comparison of the performances of different controllers in the liquid level process for a pulse change from 80 mm to 60 mm in the liquid level (regulatory problem).

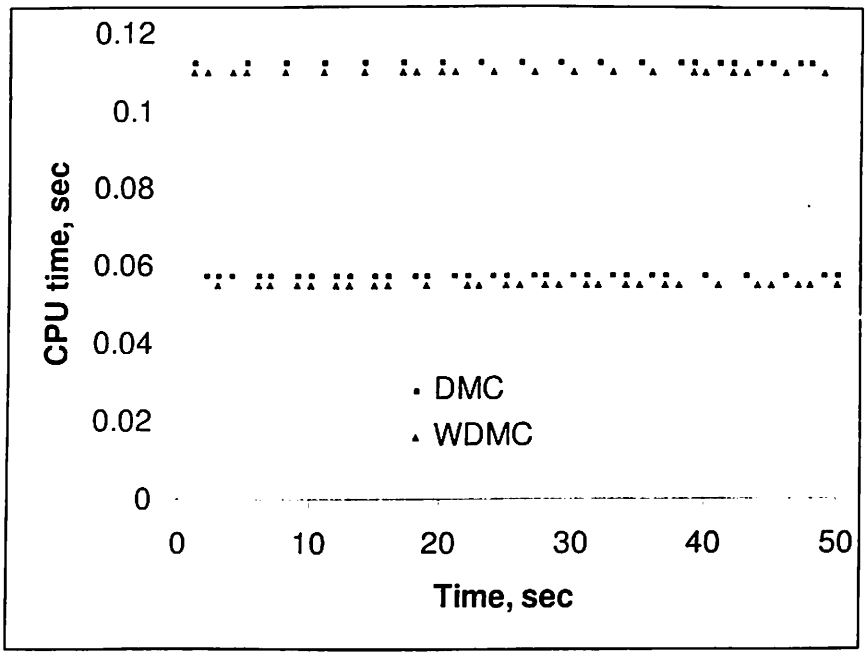


Figure 6.40 Comparison of CPU time between DMC and WDMC.

CHAPTER 7

CONCLUSIONS AND RECOMMENDATIONS

The overall conclusions drawn and the future scope of this work have been presented in this chapter.

7.1 Conclusions

Applications of wavelet-based identification and control techniques in two example processes, namely the heat exchanger process and the liquid level process have been studied through experimental investigation. Both classical least-square method and wavelet-based least-square method have been used for identification of the processes. The conventional PID, conventional DMC and Wavelet-based DMC controllers have been developed and successfully implemented in the example processes.

Based on the experimental results the following conclusions are drawn from the present study:

- The parametric model of the heat exchanger process is identified as a second order difference equation, whereas that of the liquid level process is identified as a first order difference equation.
- The wavelet-based least-square identification (WLSI) predicts better results in terms of root-mean-square-error (RMSE) than the classical least-square identification (CLSI) for both the example processes.
- The PID controller with Ziegler-Nichols (Z-N) settings gives better control action compared to the PID controller with Cohen-Coon (C-C) settings for both the processes.
- The PID controller with the settings obtained by on-line trial and error (OLTE) method gives even better performance than the PID controllers with C-C and Z-N settings in terms of minimum integral of square error (ISE).
- The conventional DMC is found to give better control action in comparison to the conventional PID controller in terms of minimum ISE for both the processes.
- WDMC takes lesser CPU time in comparison to the DMC.

- The developed PID, DMC and WDMC controllers are found to be robust as they are able to reject the moderate disturbances while maintaining the stability in case of regulatory controls.
- The developed WDMC controller outperforms the PID, DMC, FLC and the ANN-based controllers in terms of least ISE in both the example processes.

7.2 Recommendations for Future Work

- The wavelet-based identification and control has been applied to SISO (Single-Input Single-Output) processes. This can be extended to MIMO (Multi-Input Multi-Output) processes. This also can be extended to other unit operations.
- The performance of the wavelet-based controller can be compared with other advanced controllers like IMC (Internal Model Controller), QDMC (Quadratic Dynamic Matrix Controller), GLC (Globally Linearized Controllers) and non-linear PID controllers.
- For identification of the chemical processes, DAUB4 wavelet has been used. Other wavelets, like DAUB6, DAUB20, etc., may be tried for this purpose.
- For blocking and condensing, Haar wavelet is used. This work can be extended for other orthonormal wavelets.
- When wavelet functions are used in ANN structure, it is called as Wave-net or WNN (Wavelet Neural Network). This is an upcoming area of research. ANN has already been known as powerful techniques for identification and control. The combination of the two, i.e., WNN seems to provide a very good platform for process identification and control. Research can be extended in this direction.
- The wavelet-based controllers discussed here are on text-based software. A worthwhile effort will be to develop more user friendly and versatile software with visual mode for the same.
- In this work the wavelet-based controller is developed through the structure of linear model predictive controller. This can be extended for non-linear model predictive controllers (NMPC) structure.

REFERENCES

Abu-Zahra, N.H. and A. Seth, "In-process density control of extruded foam PVC using wavelet packet analysis of ultrasound waves," *Mechatronics*, 12(9-10) 1083-1095, 2002.

Adrover, A., G. Continillo, S. Crescitelli, M. Giona and L. Russo, "Wavelet-like collocation method for finite-dimensional reduction of distributed systems," *Computers & Chemical Engineering*, 24(12), 2687-2703, 2000.

Akansu, A.N. and R.A. Haddad, "Multiresolution signal decomposition: Transforms, subbands, wavelets," *Academic Press*, San Diego, CA, 1992.

Al-Assaf, Y., R. El-Khazali and W. Ahmad, "Identification of fractional chaotic system parameters," *Chaos, Solitons & Fractals*, 22(4), 897-905, 2004.

Al-Assaf, Y., "Recognition of control chart patterns using multi-resolution wavelets analysis and neural networks," *Computers & Industrial Engineering*, 47(1), 17-29, 2004.

Alpert, B.K., "Wavelets and Other Bases for Fast Numerical Linear Algebra," In "Wavelets: A Tutorial in Theory and Applications, Chui, C.K. (Ed.), *Academic Press*, New York, pp. 181-216, 1992.

Alsberg, B.K. "An introduction to wavelet transforms." http://pcf1.chembio.ntnu.no/~bka/research/waveletintro/wavetmp.html

Alvin, K.F., A. N. Robertson, G. W. Reich and K. C. Park, "Structural system identification: from reality to models," *Computers & Structures*, 81(12), 1149-1176, 2003.

Amat, S., F. Aràndiga, A. Cohen and R. Donat, "Tensor product multiresolution analysis with error control for compact image representation," *Signal Processing*, 82(4), 587-608, 2002.

Amat, S., F. Aràndiga, A. Cohen, R. Donat, G. Garcia and M. von Oehsen, "Data Compression with ENO Schemes: A Case Study," *Applied and Computational Harmonic Analysis*, 11(2), 273-288, 2001.

Antonopoulos-Domis, M. and T. Tambouratzis, "System identification during a transient via wavelet multiresolution analysis followed by spectral techniques," *Annals of Nuclear Energy*, 25(7), 465-480, 1998.

Bakshi, B.R. and G. Stephanopoulos, "Wave-Net: a Multiresolution, Hierarchical Neural Network with Localized Learning," *AIChE Journal*, 39(1), 57-81, 1993.

Balmelli, L., T. Liebling and M. Vetterli, "Computational analysis of mesh simplification using global error," *Computational Geometry*, 25(3), 171-196, 2003.

Balmelli, L., M. Vetterli and T.M. Liebling, "Mesh Optimization Using Global Error with Application to Geometry Simplification," *Graphical Models*, 64(3-4), 230-257, 2002.

Battle, G., "A block spin construction of ondelettes. Part I: Lemarie functions," *Communications in Mathematical Physics*, 110(3), 601-615, 1987.

Beylkin, G., R. Coifman, and V. Rokhlin, "Fast Wavelet Transforms and Numerical Algorithms," *Communications on Pure and Applied Mathematics*, 44, 141-183, 1991.

Bequette, B.W., "Process Control: Modeling, Design and Simulation," *Prentice-Hall of India Pvt. Ltd.*, New Delhi, 2003.

Bindal, A., J.G. Khinast, M.G. Ierapetritou, "Adaptive multiscale solution of dynamical systems in chemical processes using wavelets," *Computers and Chemical Engineering*, 27(1), 131-142, 2003.

Binder, T., A. Cruse, C.A.C. Villar and W. Marquardt, "Dynamic optimization using a wavelet based adaptive control vector parameterization strategy." *Computers & Chemical Engineering*, 24(2-7), 1201-1207, 2000.

Bokor, J., "Approximate identification for robust control," *Annual Reviews in Control*, 22, 187-198, 1998.

Bozin, A.S. and P.C. Austin, "Dynamic Matrix Control of a Paper Machine Benchmark Problem," *Control Engineering Practice*, 3(10), 1479-1482, 1995.

Briesen, H. and W. Marquardt, "Adaptive model reduction and simulation of thermal cracking of multicomponent hydrocarbon mixtures," *Computers & Chemical Engineering*, 24(2-7), 1287-1292, 2000.

Burrus, C.S., R.A. Gopinath and H. Guo, "Introduction to Wavelets and Wavelet Transforms- A Primer," *Prentice Hall*, New Jersey, 1998.

Carrier, J.F. and G. Stephanopoulos, "Multiresolution Theory for Model Identification and Control," Presented at the AIChE Annual Meeting, Miami Beach, FL, 1992.

Carrier, J.F. and G. Stephanopoulos, "Wavelet-Based Modulation in Control-Relevant Process Identification," *AIChE Journal*, 44(2), 341-360, 1998.

Cannon, M. and J. -J. E. Slotine, "Space-frequency localized basis function networks for nonlinear system estimation and control," *Neurocomputing*, 9(3), 293-342, 1995.

Chen, B.H., J.M. Woodley, "Wavelet shrinkage data processing for neural networks in bioprocess modeling," *Computers & Chemical Engineering*, 26(11) 1611-1620, 2002.

Chidambaram, M., "Computer Control of Processes," *Nasosa Publishing House*, New Delhi, 2002.

Chikkula, Y. and J. H. Lee, "Application of Wavelets in Process Control," In "Wavelet Applications in Chemical Engineering," Motard, R. L. and B. Joseph (Eds.), *Kluwer Academic Publishers*, Boston, 1994.

Chui, C.K., and J.Z. Wang, "On Compactly Supported Spline Wavelets and a Duality Principle," *Transactions of the American Mathematical Society*, 330(2), 903-915, 1992.

Chui, C.K., L. Montefusco and L. Puccio, (Eds.), "Wavelets: Theory, Algorithms, and Applications," *Academic Press*, San Diego, 1994.

Coca, D. and S.A. Billings, "Non-linear system identification using wavelet multiresolution models," *International Journal of Control*, 74(18), 1718-1736, 2001.

Cohen, G.H. and G.A. Coon, "Theoretical Considerations of Retarded Control," *Transactions of the American Society for Mechanical Engineers*, 75, 827, 1953.

Cohen, A., I. Daubechies, and B. Jawerth, and P. Vial, "Multiresolution Analysis, Wavelets and Fast Algorithms on an Interval," *C.R. Acad. Sci. Paris*, 316(I), 417-421, 1993.

Coifman, R., and M.V. Wickerhauser, "Entropy-Based Algorithms for Best Basis Selection," *IEEE Transactions on Information Theory*, 38(2), 713-718, 1992.

Coughanowr, D.R., "Process Systems Analysis and Control," 2nd ed., *McGraw-Hill International Editions*, Singapore, 1991.

Cruz, P., A. Mendes, and F. D. Magalhaes, "Using wavelets for solving PDEs: An adaptive collocation method," *Chemical Engineering Science*, 56(10), 3305-3309, 2001.

Cruz, P., A. Mendes and F. D. Magalhaes, "Wavelet-based adaptive grid method for the resolution of nonlinear PDEs," *AIChE Journal*, 48(4), 774-785, 2002.

Cutler, C.R., and B.L. Ramaker, "Dynamic Matrix Control-A Computer Control Algorithm." In *Proceedings of the Joint Automatic Control Conference*, San Francisco, CA, August 13-15, 1980, paper WP5-B. Also presented at the AIChE National Meeting, Houston, TX, 1979.

Cutler, C.R., "Dynamic Matrix Control: An Optimal Multivariable Control Algorithm with Constraints," *Ph.D. Thesis*, University of Houston, Houston, TX, 1983.

Dasgupta, M.S. and R.K. Gupta, "Implementation of Fuzzy Control on A Heat Exchanger," *Indian Chemical Engineer: Section A*, 40(4), 336-346, 1998.

Dasgupta, M.S., "Identification and Control of Some Unit Operations Using Fuzzy Logic and Neural Network," *Ph.D. Thesis*, Birla Institute of Technology and Science, Pilani (Rajasthan), India, 1999.

Dasgupta, M.S., "Fuzzy and ANN Controller Design and Implementation on a Level Control Setup." In *Proceedings of the UGC National Conference on Advances in Industrial Automation*, 76-82, Jodhpur (Rajasthan), India, March 20-21, 2004.

Daubechies, I., "Orthonormal bases of compactly supported wavelets," *Communications on Pure and Applied Mathematics*, 41(7), 909-996, 1988.

Daubechies, I., "The wavelet transform: a method for time-frequency localization," Advances in Spectrum Analysis and Array Processing, Volume I, Haykin, S. (Ed.), *Prentice Hall*, Englewood Cliffs, NJ, 366-417, 1991.

Daubechies, I., "Ten Lectures in Wavelets," CBMS-NSF Regional Conference Series in Applied Mathematics, Vol. 61, *Society for Industrial and Applied Mathematics* (SIAM), Philadelphia, Pennsylvania, 1992.

Daubchies, I., "The wavelet transform, time-frequency localization and signal analysis," *IEEE Transactions on Information Theory*, 36(5), 961-1005, 1990.

Daubchies, I., "Wavelets Making Waves in Mathematics and Engineering," Videotape, *American Mathematical Society*, 1993.

Daubechies, I., S. Mallat and A.S. Willsky (Eds.), "Special Isssue on Wavelet Transforms and Multiresolution Signal Analysis," *IEEE Transactions on Information Theory*, 38(2), Part 2, 1992.

Daubechies, I., A. Grossmann and Y. Meyer, "Painless non-orthogonal expansions," *Journal of Mathematical Physics*, 27(1), 293-309, 1986.

Daubechies, I. and A. Grossmann, "Frames in the Bargmann space of Entire-functions," *Communications on Pure and Applied Mathematics*, 41(2), 151-164, 1988.

Dorfan, Y., A. Feuer and B. Porat, "Modeling and identification of LPTV systems by wavelets," *Signal Processing*, 84(8), 1285-1297, 2004.

Doroslovacki, M., H. Fan and L. Yao, "Wavelet-Based Identification of Linear Discrete-Time Systems: Robustness Issue," *Automatica*, 34(12), 1637-1640, 1998.

Dracopoulos, D. C., "Evolutionary Learning Algorithms for Neural Adaptive Control," *Springer-Verlag*, London, 1997.

Edgar, T.F. and D.M. Himmelblau, "Optimization of Chemical Processes," *McGraw-Hill Book Company*, Singapore, International edition, 1989.

Fadili, M. J. and E. T. Bullmore, "Wavelet-Generalized Least Squares: A New BLU Estimator of Linear Regression Models with 1/f Errors," *NeuroImage*, 15(1), 217-232, 2002.

Feng, W., H. Genceli and M. Nikolaou, "Constrained model predictive control with simultaneous identification using wavelets," *Computers & Chemical Engineering*, 20, Supplement 2, S1011-S1016, 1996.

Gao, Y., Q. Chen, S. Tse and X. Xu, "A wavelet approach to determine the switching frequency for composite control during surface grinding," *Journal of Materials Processing Technology*, 129(1-3), 480-484, 2002.

Garcia, C.E. and M. Morari, "Internal Model Control. 1. A Unifying Review and Some New Results," *Industrial & Engineering Chemistry Process Design and Development*, 21(2), 308-323, 1982.

Georgiou, A., C. Georgakis and W.L. Luyben, "Nonlinear Dynamic Matrix Control for High-Purity Distillation Columns," *AIChE Journal*, 34(8), 1287-1298, 1988.

Ghanem, R. and F. Romeo, "A wavelet-based approach for model and parameter identification of non-linear systems," *International Journal of Non-Linear Mechanics*, 36(5), 835-859, 2001.

Ghanem, R. and F. Romeo, "A Wavelet-Based Approach For The Identification Of Linear Time-Varying Dynamical Systems," *Journal of Sound and Vibration*, 234(4), 555-576, 2000.

Goswami, J.C. and A.K. Chan, "Fundamentals of Wavelets: Theory, Algorithms, and Applications," *John Wiley & Sons*, New York, 1999.

Goumas, S., M. Zervakis, A. Pouliezos and G. S. Stavrakakis, "Intelligent on-line quality control of washing machines using discrete wavelet analysis features and likelihood classification," *Engineering Applications of Artificial Intelligence*, 14(5), 655-666, 2001.

Goupillaud, P., A. Grossmann, and J. Morlet, "Cycle-octave and related transforms in seismic signal analysis," *Geoexploration*, 23(1), 85-102, 1984a.

Goupillaud, P., A. Grossmann, and J. Morlet, "Cycle-octave representation for instantaneous frequency-spectra," *Geophysics*, 49(5), 669, 1984b.

Graps, A., "An Introduction to Wavelets", 1995-2004. http://www.amara.com/IEEEwave/IEEEwavelet.html.

Feng, W., H. Genceli and M. Nikolaou, "Constrained model predictive control with simultaneous identification using wavelets," *Computers & Chemical Engineering*, 20, Supplement 2, S1011-S1016, 1996.

Gao, Y., Q. Chen, S. Tse and X. Xu, "A wavelet approach to determine the switching frequency for composite control during surface grinding," *Journal of Materials Processing Technology*, 129(1-3), 480-484, 2002.

Garcia, C.E. and M. Morari, "Internal Model Control. 1. A Unifying Review and Some New Results," *Industrial & Engineering Chemistry Process Design and Development*, 21(2), 308-323, 1982.

Georgiou, A., C. Georgakis and W.L. Luyben, "Nonlinear Dynamic Matrix Control for High-Purity Distillation Columns," *AIChE Journal*, 34(8), 1287-1298, 1988.

Ghanem, R. and F. Romeo, "A wavelet-based approach for model and parameter identification of non-linear systems," *International Journal of Non-Linear Mechanics*, 36(5), 835-859, 2001.

Ghanem, R. and F. Romeo, "A Wavelet-Based Approach For The Identification Of Linear Time-Varying Dynamical Systems," *Journal of Sound and Vibration*, 234(4), 555-576, 2000.

Goswami, J.C. and A.K. Chan, "Fundamentals of Wavelets: Theory, Algorithms, and Applications," *John Wiley & Sons*, New York, 1999.

Goumas, S., M. Zervakis, A. Pouliezos and G. S. Stavrakakis, "Intelligent on-line quality control of washing machines using discrete wavelet analysis features and likelihood classification," *Engineering Applications of Artificial Intelligence*, 14(5), 655-666, 2001.

Goupillaud, P., A. Grossmann, and J. Morlet, "Cycle-octave and related transforms in seismic signal analysis," *Geoexploration*, 23(1), 85-102, 1984a.

Goupillaud, P., A. Grossmann, and J. Morlet, "Cycle-octave representation for instantaneous frequency-spectra," *Geophysics*, 49(5), 669, 1984b.

Graps, A., "An Introduction to Wavelets", 1995-2004. http://www.amara.com/IEEEwave/IEEEwavelet.html.

Grossmann, A. and J. Morlet, "Decomposition of hardy functions into square integrable wavelets of constant shape," *SIAM Journal on Mathematical Analysis*, 15(4), 723-736, 1984.

Grossmann, A., J. Morlet, and T. Paul, "Transforms associated to square integrable group - Representations 1: General results," *Journal of Mathematical Physics*, 26(10), 2473-2479, 1985.

Grossmann, A., J. Morlet, and T. Paul, "Transforms associated to square integrable group - Representations 2: Examples," *Annales de l'Institut Henri Poincare-physique Theorique*, 45(3), 293-309, 1986.

Grossmann, A., M. Holschneider, R. Kronland-Martinet, and J. Morlet, "Detection of abrupt changes in sound signals with the help of wavelet transforms," *Advances in Electronics and Electron Physics*, Supplement 19, 289-306, 1987.

Gu, D. and H. Hu, "Neural predictive control for a car-like mobile robot," *Robotics and Autonomous Systems*, 39(2), 73-86, 2002.

Gupta, S.K., "Numerical Methods for Engineers," *Wiley Eastern Ltd.*, New Delhi, 1995.

Güttler, S. and H. Kantz, "The Auto-Synchronized Wavelet Transform Analysis for Automatic Acoustic Quality Control," *Journal of Sound and Vibration*, 243(1), 3-22, 2001.

Guan, S., C. -H. Lai and G. W. Wei, "Geometry and boundary control of pattern formation and competition," *Physica D: Nonlinear Phenomena*, 176(1-2), 19-43, 2003.

Haar, A., "Zur Theorie der orthogonalen Funktionen-Systeme," *Mathematics Annuals*, 69, 331-371, 1910.

Hasiewicz, Z. and M. Pawlak, "Nonlinear System Identification via Wavelet Expansions," *IFAC System Identification*, Santa Barbara, California, U.S.A, 2000.

Haykin, S., "Neural Networks: A Comprehensive Foundation," 2nd Ed., *Prentice Hall*, Englewood Cliffs, N.J., USA, 1998.

Hemeida, A. M., "Wavelet neural network load frequency controller," *Energy Conversion and Management*, 46(9-10), 1613-1630, 2005.

Hernández, E. and G. Weiss, "A First Course on Wavelets," *CRC Press*, Boca Raton, 1996.

Huntsberger, T. and J. Rose, "BISMARC: a biologically inspired system for map-based autonomous rover control," *Neural Networks*, 11(7-8), 1497-1510, 1998.

Jawerth, B. and W. Sweldens, "An overview of wavelet based multiresolution analysis," Research report 1993 (1), Department of Mathematics, *University of South Calolina*, 1993.

Johnson, A. E. and M. Hebert, "Control of Polygonal Mesh Resolution for 3-D Computer Vision," *Graphical Models and Image Processing*, 60(4), 261-285, 1998.

Juditsky, A., H. Hjalmarsson, A. Benveniste, B. Delyon, L. Ljung, J. SjÖberg and Q. Zhang, "Nonlinear black-box models in system identification: Mathematical foundations," *Automatica*, 31(12), 1725-1750, 1995.

Kaiser, G., "A Friendly Guide to Wavelets," *Birkhäuser*, Boston, 1994.

Kaufhold, B., R. L. Kirlin and R. M. Dizaji, "Blind System Identification Using Normalized Fourier Coefficient Gradient Vectors Obtained from Time-Frequency Entropy-Based Blind Clustering of Data Wavelets," *Digital Signal Processing*, 9(1), 18-35, 1999.

Kavchak, M. and H. Budman, "Adaptive neural network structures for non-linear process estimation and control," *Computers & Chemical Engineering*, 23(9), 1209-1228, 1999.

Kijewski, T. L. and A. Kareem, "Wavelet Transforms for System Identification and Associated Process Concerns," 15th ASCE Engineering Mechanics Conference Columbia University, New York, June 2-5, 2002.

Knapp, T. D., H. M. Budman and G. Broderick, "Adaptive control of a CSTR with a neural network model," *Journal of Process Control*, 11(1), 53-68, 2001.

Kosanovich, K. A., A. R. Moser and M. J. Piovoso, "Poisson wavelets applied to model identification," *Journal of Process Control*, 5(4), 225-234, 1995.

Kosanovich, K. A., A. R. Moser and M. J. Piovoso, "A new family of wavelets: the Poisson wavelet transform," *Computers & Chemical Engineering*, 21(6), 601-620, 1997.

Le, T. -P. and P. Argoul, "Continuous wavelet transform for modal identification using free decay response," *Journal of Sound and Vibration*, 277(1-2), 73-100, 2004.

Lee, J. H. and Z. H. Yu, "Tuning of Model Predictive Controllers for Robust Performances," *Computers & Chemical Engineering*, 18(1), 15-37, 1994.

Lee, J. H., Y. Chikkula, Z. H. Yu and J. C. Kantor, "Improving the Computational Efficiency of the Model Predictive Control Algorithm Using the Wavelet Transformation," *International Journal of Control*, 61, 859-883, 1992.

Lepik, Ü., "Numerical solution of differential equations using Haar wavelets," *Mathematics and Computers in Simulation*, 68(2), 127-143, 2005.

Liandrat, J., V. Perrier and P. Tchamitchian, "Numerical Resolution of Nonlinear Partical Differential Equations Using the Wavelet Approach," In "Wavelets and their Applications", M. B. Ruskai (Ed.), 227-238, *Jones and Bartlett Publishers*, Boston, MA, 1992.

Liu, Y. and I. T. Cameron, "A new wavelet-based method for the solution of the population balance equation," *Chemical Engineering Science*, 56(18), 5283-5294, 2001.

Liu, Y., I. T. Cameron and F. Y. Wang, "The wavelet-collocation method for transient problems with steep gradients," *Chemical Engineering Science*, 55(9), 1729-1734, 2000.

Ljung, L., "System Identification," Chapter 58 (pp. 1033-1054) in *The Control Handbook*, W.S. Levine (Ed.), *CRC Press*, 1996.

Luo, G. Y., D. Osypiw and M. Irle, "Surface quality monitoring for process control by on-line vibration analysis using an adaptive spline wavelet algorithm," *Journal of Sound and Vibration*, 263(1), 85-111, 2003.

Luyben, W.L., "Process Modeling, Simulation and Control for Chemical Engineers," 2nd Ed., *McGraw-Hill Publishing Company*, New York, 1990.

Luyben, M.L. and W.L. Luyben, "Essentials of Process Control," *The McGraw-Hill Companies*, New York, 1997.

Mahadevan, N. and K. A. Hoo, "Wavelet-based model reduction of distributed parameter systems," *Chemical Engineering Science*, 55(19), 4271-4290, 2000.

Maiti, S.N. and D.N. Saraf, "Adaptive dynamic matrix control of a distillation column with closed-loop online identification," *Journal of Process Control*, 5(5), 315-327, 1995.

Mallat, S., "A theory for multiresolution signal decomposition," *Dissertation*, University of Pennsylvania, Department of Electrical Engineering and Computer Science, 1988.

Mallat, S., "A Theory of Multiresolution Signal Decomposition: The Wavelet Representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7), 674-693, 1989.

Mallinson, S. G., J. A. Reizes, G. Hong and P. S. Westbury, "Analysis of hot-wire anemometry data obtained in a synthetic jet flow," *Experimental Thermal and Fluid Science*, 28(4), 265-272, 2004.

Marchant, B. P., "Time-frequency Analysis for Biosystems Engineering," *Biosystems Engineering*, 85(3), 261-281, 2003.

Masson, P. and A. Berry, "Comparison Of Several Strategies In The Active Structural Acoustic Control Using Structural Strain Measurements," *Journal of Sound and Vibration*, 233(4), 703-722, 2000.

Maurath, P.R., D.A. Mellichamp and D.E. Seborg, "Predictive Controller Design for Single-Input/Single-Output (SISO) Systems," *Industrial & Engineering Chemistry Research*, 27(6), 956-963, 1988.

Meyer, Y., "Orthonormal wavelets." In *Proceedings of the 9th International Congress on Mathematical Physics*, University College of Swansea, Swansea, July 17-27, 1988. B. Simon, A. Truman and I. M. Davis (Eds.), *Adam Hilger Ltd.*, Bristol, 1989.

Misra, M., S. Kumar, S. J. Qin and D. Seemann, "Error based criterion for on-line wavelet data compression," *Journal of Process Control*, 11(6), 717-731, 2001.

Morari, M. and E. Zafiriou, "Robust Process Control," *Prentice-Hall*, Englewood Cliffs, N.J, U.S.A., 1989.

Morlet, J., G. Arens, I. Fourgeau and D. Giard, "Wave propagation and sampling theory," *Geophysics*, 47(2), 203-236, 1982.

Nikolaou, M. and Y. You, "Solution of Partial Differential Equations Using Wavelets" Presented at the AIChE Annual Meeting, Miami Beach, FL, 1992.

Nikolaou, M. and Y. You, "Use of Wavelets for Numerical Solution of Differential Equations" In "Wavelet Applications in Chemical Engineering," R. L. Motard, and B. Joseph (Eds.), *Kluwer Academic Publishers*, Boston, 1994.

Nikolaou, M. and P. Vuthandam, "FIR Model Identification: Parsimony Through Kernel Compression with Wavelets," *AIChE Journal*, 44(1), 141-150, 1998.

Ogden, T. and E. Parzen, "Data dependent wavelet thresholding in nonparametric regression with change-point applications," *Computational Statistics & Data Analysis*, 22(1), 53-70, 1996.

Ogunnaike, B. A. and W. H. Ray, "Process dynamics, modeling and control," *Oxford University Press*, New York, 1994.

Ogunnaike, B.A., "Dynamic Matrix Control: A Nonstochastic, Industrial Process Control Technique with Parallels in Applied Statistics," *Industrial & Engineering Chemistry Research Fundamentals*, 25(4), 712-718, 1986.

Oysal, Y., A. S. Yilmaz and E. Koklukaya, "A dynamic wavelet network based adaptive load frequency control in power systems," *International Journal of Electrical Power & Energy Systems*, 27(1), 21-29, 2005.

Palavajjhala, S., R. L. Motard and B. Joseph, "Blocking and Condensing Design for Quadratic Dynamic Matrix Control using Wavelets," *Industrial & Engineering Chemistry Research*, 33(5), 1159-1173, 1994.

Palavajjhala S., R. L. Motard and B. Joseph, "Process Identification Using Discrete Wavelet Transforms: Design of Prefilters," *AIChE Journal*, 42(3), 777-790, 1996.

Palavajjhala, S., R.L. Motard, and B. Joseph, "Computational Aspects of Wavelets and Wavelet Transforms" In "Wavelet Applications in Chemical Engineering," Motard, R. L. and B. Joseph (Eds.), *Kluwer Academic Publishers*, Boston, 1994.

Papoulis, A., "Signal Analysis", *McGraw-Hill*, New York, 1977.

Pati, Y., R. Rezaiifar, P. Krishnaprasad and W. Dayawansa, "A fast recursive algorithm for system identification and model reduction using rational wavelets." In *Proceedings of the 27th Asilomar Conference on Signals, Systems and Computers*, Vol. 1, p.35, Pacific Grove, CA., November 1993.

Piombo, B. A. D., A. Fasana, S. Marchesiello and M. Ruzzene, "Modelling And Identification of The Dynamic Response of a Supported Bridge," *Mechanical Systems and Signal Processing*, 14(1), 75-89, 2000.

Pislaru, C., J. M. Freeman and D. G. Ford, "Modal parameter identification for CNC machine tools using Wavelet Transform," *International Journal of Machine Tools and Manufacture*, 43(10), 987-993, 2003.

Polikar, R., "The Wavelet Tutorial - Part 1: Fundamental Concepts and an Overview of the Wavelet Transform", Second Edition, 1996a. http://users.rowan.edu/~polikar/WAVELETS/WTpart1.html

Polikar, R., "The Wavelet Tutorial - Part 2 Fundamentals: The Fourier Transform and The Short Term Fourier Transform, Resolution Problems" 1996b. http://users.rowan.edu/~polikar/WAVELETS/WTpart2.html.

Poterasu, V. F., "Wavelets transform for nonlinear control of multibody systems," *Journal of the Franklin Institute*, 338(2-3), 321-334, 2001.

Qiao, F. and R. Milam, "Smoothness and Vanishing Wavelet Moments, Version 2.2: 2003/06/03 15:39:33.922 GMT-5," in the Internet site http://cnx.rice.edu/content/m11172/latest/.

Press, W. H., S. A. Teukolsky, W. T. Vetterling and B. P. Flannery, Numerical Recipes in C: The Art of Scientific Computing, 2nd Ed., *Cambridge University Press*, 1992.

Rao, R.M. and A.S. Bopardikar, "Wavelet Transforms: Introduction to Theory and Applications," *Pearson Education Asia*, Delhi, 1998.

Rawlings, J. B. and K. R. Muske, "The Stability of Constrained Receding Horizon Control," *IEEE Transactions on Automatic Control*, 38(10), 1512-1516, 1993.

Razzaghi, M. and S. Yousefi, "Legendre wavelets method for the solution of nonlinear problems in the calculus of variations," *Mathematical and Computer Modelling*, 34(1-2), 45-54, 2001.

Russell, B. D., J. Lasenby, S. Blackburn and D. I. Wilson, "Characterising paste extrusion behaviour by signal processing of pressure sensor data," *Powder Technology*, 132(2-3), 233-248, 2003.

Safavi, A. A., A. Nooraii and J. A. Romagnoli, "A hybrid model formulation for a distillation column and the on-line optimisation study," *Journal of Process Control*, 9(2), 125-134, 1999.

Safavi, A. A. and J. A. Romagnoli, "Application of wavelet-based neural networks to the modelling and optimisation of an experimental distillation column," *Engineering Applications of Artificial Intelligence*, 10(3), 301-313, 1997.

Saraf, D.N. and S. Ganguly, "Research in Advanced Control and Online Optimization for Chemical Process Industry," *Indian Chemical Engineer: Section A*, 39(3), 176-183, 1997.

Schoenwald, D. A., "System Identification Using a Wavelet-Based Approach." In *Proceedings of the 32nd Conference on Decision Control*, San Antonio, Texas, December 15-17, 1993.

Shieh, N. -C., C. -T. Chang, C. -L. Lin and H. –Y. Jan, "Robust position control of a transportation carriage directly driven by linear motor using wavelet neural," *Engineering Applications of Artificial Intelligence*, 15(5), 479-489, 2002.

Shinbrot, M., "On the Analysis of Linear and Nonlinear Systems," *Transactions of the American Society of Mechanical Engineers*, 79, 547, 1957.

Shmilovici, A. and O. Z. Maimon, "The Fuzzy Rule-Base Solution of Differential Equations," *Information Sciences*, 92, 233-254, 1996.

Shmilovici, A. and O. Z. Maimon, "On the solution of differential equations with fuzzy spline wavelets," *Fuzzy Sets and Systems*, 96(1), 77-99, 1998.

Sjöberg, J., Q. Zhang, L. Ljung, A. Benveniste, B. Delyon, P. -Y. Glorennec, H. Hjalmarsson and A. Juditsky, "Nonlinear black-box modeling in system identification: a unified overview," *Automatica*, 31(12), 1691-1724, 1995.

Shridhar, R. and D.J. Cooper, "A Tuning Strategy for Unconstrained SISO Model Predictive Control," *Industrial and Engineering Chemistry Research*, 36(3), 729-746, 1997.

Srivastava, S., M. Singh, M. Hanmandlu and A.N. Jha, "New fuzzy wavelet neural networks for system identification and control," *Applied Soft Computing*, 6(1), 1-17, 2005.

Sripathi, D., "Efficient Implementation of Discrete Wavelet Transforms using FPGAs," *M.S. Thesis*, The Florida State University College of Engineering, 2003.

Staszewski, W. J., "Identification of Non-Linear Systems Using Multi-Scale Ridges and Skeletons of the Wavelet Transform," *Journal of Sound and Vibration*, 214(4), 639-658, 1998.

Stephanopoulos, G. and J. F. Carrier, "Generation and Validation of Models for the Design of Process Controllers" Presented at the AIChE Annual Meeting, Los Angeles, CA, 1991.

Stephanopoulos, G., "Chemical Process Control: An Introduction to Theory and Practice," *Prentice-Hall of India Pvt. Ltd.*, New Delhi, 1993.

Stromberg, J.O., "A modified Franklin system and higher order spline systems on R^n as unconditional bases for hardy spaces," Conference in Honour of A. Zygmund, Becker *et al.* (eds), Vol. II, 475-493, 1981.

Sun, Y., N. Xi and J. Tan, "On-line Parameter Identification of a Cart by Mobile Manipulation Pushing," *Robotics and Autonomous Systems*, 46(1), 29-46, 2004.

Suter, B.W. and M.E. Oxley, "Getting Around the Balian-Low Theorem Using Generalized Malvar Wavelets," In "Wavelets: Theory, Algorithms, and Applications," Chui, C.K., L. Montefusco, and L. Puccio (Eds.), 295-310, *Academic Press*, San Diego, 1994.

Takahashi, M., H. Ohmori and A. Sano, "Impulse response identification by use of wavelet packets decomposition." In *Proceedings of the IEEE Conference on Decision and Control*, 1, 211, 1998.

Temme, N.M., "Wavelets: First Steps" In "Wavelets: An Elementary Treatment of Theory and Applications," Koornwinder, T. H. (Ed.), *World Scientific Publishing Co. Pte. Ltd.*, Singapore, 1993.

Tsatsanis, M. K. and G. B. Giannakis, "Time-Varying System Identifaication and Model Validation Using Wavelets," *IEEE Transactions on Signal Processing*, 41(12), 3512-3523, 1993.

Valens, C., "A Really Friendly Guide to Wavelets, Part 1, 1999.

http://perso.wanadoo.fr/ polyvalens/clemens/wavelets/wavelets.html

Wai, R. -J. and J. -M. Chang, "Intelligent control of induction servo motor drive via wavelet neural network," *Electric Power Systems Research*, 61(1), 67-76, 2002.

Walker, J.S., "A Primer on Wavelets and their Scientific Applications," *Chapman & Hall/CRC*, New York, 1999.

Wang, Y. -M., S. -Il Kwon, A. Regan, and T. Rohlev, "System Identification of the Linac RF System Using A Wavelet Method and Its Applications in the SNS LLRF Control System." In *Proceedings of the Particle Accelerator Conference*, Chicago, USA, June 18-22, 2001.

Wang, D., J. A. Romagnoli and A. A. Safavi, "Wavelet-Based Adaptive Robust M-Estimator for Nonlinear System Identification," *AIChE Journal*, 46(8), 1607-1615, 2000.

Wang, Y. and K. S. Moon, "A methodology for the multi-resolution simulation of grinding wheel surface," *Wear*, 211(2), 218-225, 1997.

Wickerhauser, M.V., "Lectures on Wavelet Packet Algorithms," preprint, Washington University, St Louis, 1991.

Xin, J. and A. Sano, "Adaptive system identification based on generalized wavelet decomposition," *Applied Mathematics and Computation*, 69(1), 97-109, 1995.

Yang, Z. -J., S. Sagara and T. Tsuji, "System Impulse Response Identification Using a Multiresolution Neural Network," *Automatica*, 33(7), 1345-1350, 1997.

Zhang, Q. and A. Benveniste, "Wavelet Networks" *IEEE Transactions on Neural Networks*, 3(6), 889-898, 1992.

Zhao, G., S. Xu, W. Li and O. E. Teo, "Fast variational design of multiresolution curves and surfaces with B-spline wavelets," *Computer-Aided Design*, 37(1), 73-82, 2005.

Zhou, Y. -H., J. Wang, X. J. Zheng and Q. Jiang, "Vibration Control Of Variable Thickness Plates with Piezoelectric Sensors and Actuators Based on Wavelet Theory," *Journal of Sound and Vibration*, 237(3), 395-410, 2000.

Zhou, Y. -H. and J. Wang, "Vibration control of piezoelectric beam-type plates with geometrically nonlinear deformation," *International Journal of Non-Linear Mechanics*, 39(6), 909-920, 2004.

Ziegler, J.G. and N.B. Nichols, "Optimum Settings for Automatic Controllers," *Transactions of the American Society of Mechanical Engineers*, 64, 759, 1942.

APPENDIX I

COMPUTER PROGRAMS

The following computer codes in C language are developed for identification and online control of the heat exchanger process. These are also used for identification and control of the liquid level process with minor modifications. Some longer lines in the programs are broken due to limitation in width of the paper.

List of Programs **Page No.**

1. Source code in C for accessing ADC card. Program name: *ADC.C*

```
#include <stdio.h>
#include <stdlib.h>
#include <dos.h>
#include <conio.h>
#include <time.h>
#define BASE 0x218
void main()
{
  int lb, hb, status, adc_no, i;
  float temp, voltage;
  double el_t = 0.0;
  time_t init, final;
  clrscr();
  while(!kbhit())
  {
    init = time(NULL);
    voltage = 0.0;
    for (i=0; i<2000; i++)
    {
        outportb(BASE+0, 0);
        outportb(BASE+1, 0);
        status = inport (BASE+0);
        while (status < 128)
          status = inport (BASE+0);
        lb = inportb(BASE+2);
        hb = inportb(BASE+3);
        hb = hb & 0x0f;
        adc_no = lb + 256*hb;
        voltage = voltage + (adc_no/4096.0)*10 - 5.0;
    }
    delay(1000);
    final = time(NULL);
    el_t = el_t + difftime (final,init);
    temp = (voltage/2000.0)*100.0; /* 0 to 1 Volt is converted to 0 to 100 °C. */
    printf("Time=%lf\t Voltage=%f\t Temp.=%f\n", el_t, voltage, temp);
  }
}
```

2. Source code in C for accessing DAC card. Program name: *DAC.C*

/* This Program converts the controller output (CO), which is a value between 0 and 4 to corresponding voltage between 0 and 1 Volt. This value is to be converted to the current signal (4-20mA) using electrical console before sending to the final control element. */

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
```

```
#include<conio.h>
#include<dos.h>
#include<time.h>
#define BASE 0x218

void main()
{
        int n,lb,hb;
        float v0;
        printf("Enter the initial controller output (CO) in the range 0 to 4: ");
        scanf("%f",&v0);
        while(!kbhit())
        {
                n = (int) ((v0+5.0)*4095.0/10);
                lb = n & 0xff;
                hb = (int) (n-lb)/256.0;
                outportb (BASE + 2, lb);
                outportb (BASE + 3, hb);
        }
        getch();
        clrscr();
}
```

3. Source code in C for obtaining input-output data in the heat exchanger process. Program name: *IODATA.C*

```
/*Source code for finding the input output relationship of the heat exchanger */
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<conio.h>
#include<dos.h>
#include<time.h>
#define BASE 0x218
FILE *fp;
time_t init,final;
void steady_state();
void send_output(float v0);
float read_temp();
void main()
{
        float c_pos, lower, upper,a_time=0.0, temp;
        clrscr();
        lower = 0.5;  /*Lower controller position*/
        upper = 2.5;  /*Upper controller position*/
        fp=fopen("c:\\tc3\\bin\\tioexpt6.txt","a+");
        steady_state();
        //if(!fp) exit(-1);
        while(!kbhit())
        {
```

```
                printf("Check connections before hitting any key\n");
                printf("Hit any key when steady state is reached\n\n\n");
                temp = read_temp();
                printf("Temperature=%f\n",temp);
                delay(2000);
        }
        getch();
        clrscr();
        printf("PROGRAME STARTS - DON'T HIT ANY KEY\n");
        fprintf(fp, "\nAll glories to Sri Sri Guru and Gauranga\n");
        fprintf(fp, "DATE: 27TH SEPT 2004\n");
        fprintf(fp, "Input output relationship in h/e expt- file: tio_expt.c\n");
        fprintf(fp, "Cold water flow rate is constant at 200ml/min\n");
        fprintf(fp, "Hot water inlet temp is const at around 80 +- 3 deg C\n");
        fprintf(fp, "Controller Position between %f and %f\n\n", lower, upper);
        fprintf(fp,"\nTime\tController Position\tTemperature(deg. C)\n");
        while(!kbhit())
        {
                init = time(NULL);
                /*Random selection of the controller output between 0.5 and 2.5 */
                c_pos =lower +((float)rand())/((float)RAND_MAX)*(upper-lower);
                /* Generation of PRBS (Pseuso Ramdom Binary Sequence) */
                if(c_pos>=1.0)
                {
                        c_pos = 2.5;
                }
                else
                {
                        c_pos = 0.5;
                }
                send_output(c_pos);
                delay(1000);
                temp = read_temp();
                final = time(NULL);
                a_time += difftime(final,init);
                printf("Time=%f\tC_Position=%f\tTemp=%f\n",a_time,c_pos,temp);
                fprintf(fp,"%f\t %f\t %f\n",a_time,c_pos,temp);
        }
        fclose(fp);
}
void steady_state()
{
        int n,lb,hb;
        float v0;
        printf("Enter the initial controller position: ");
        scanf("%f",&v0);
        printf("HARE KRISHNA !\n");
        printf("<HIT ANY KEY TO START>\n");
    fprintf(fp,"HARE KRISHNA\n");
        fprintf(fp,"Initial controller position=%f\n",v0);
        while(!kbhit())
```

```
	{
		n = (int) ((v0+5.0)*4095.0/10);
		lb = n & 0xff;
		hb = (int) (n-lb)/256.0;
		outportb (BASE + 2, lb);
		outportb (BASE + 3, hb);
	}
	getch();
	clrscr();
}
void send_output(float v0)
{
	int n,lb,hb;
	n = (int) ((v0 + 5.0)*4095.0/10.0);
	lb = n & 0xff;
	hb = (int) (n - lb)/256.0;
	outportb (BASE + 2, lb);
	outportb (BASE + 3, hb);
}
float read_temp()
{
	int i,lb,hb,status,adc_no;
	float temp,voltage=0.0;

	for(i=0; i<2000; i++)
	{
		outportb (BASE+0, 0);
		outportb (BASE+1, 0);
		status = inport (BASE+0);
		while(status < 128)
			status = inport (BASE+0);
		lb = inportb (BASE+2);
		hb = inportb (BASE+3);
		hb = hb & 0x0f;
		adc_no = lb + 256*hb;
		voltage += (adc_no/4096.0)*10.0 - 5.0;
	}
	temp = (voltage/2000.0)*100.0;
	return temp;

}
```

4. Source code in C for classical least-square identification. Program name: *CLSI.C*

```
/*Process identification using least-square method.*/
#include<stdio.h>
#include<math.h>

double **get_matrix(double **,int n, int m);
double *get_vector(double *,int n);
```

```
void init_matrix(double **,int n, int m);
void unit_matrix(double **, int n);
void print_matrix(double **,int n, int m);
void ludcmp(double **, double **, int n);
void luforwdsb(double **, double *, int n);
void lubksb(double **, double *, int n);
double  **ainv_b(double **a, double **matb, int n, int m);
double **matrix_mult(double **a, int r1,int c1,double **b,int r2,int c2);
double **matrix_tranps(double **a, int nrows,int ncols);
double **least_square(double **a, int m, int n, double **x);

FILE *fp1,*fp2, *fp3;
main()
{
        int i,j,k,m,n,o,p,ct,nt,np,fc;
        double **a, **x, **paramtrs,*xdata, *mdata, **xmodel, *xdatamod, sum;

        /*Here A is the matrix of output and manipulated variables.
        X is a vector of output variables. Here it is taken as a column
        matrix for ease of matrix multiplication. Similarly PARAMTRS is also
        a vector of parameters though it is taken as a matrix.*/
        clrscr();
        p=900; //Total number of data

        nt=80; //  truncation horizon

        o=1;     //order of model

        np=10; //length of prediction horizon


        ct=nt-1;//current time instant, ct ( should be >=nt-1)

        m=nt-o+1; /* m and n are the dimensions of matrix A for least sqr.*/
        n=2*o;

        xdata = (double *)malloc(sizeof(double) * p); //Controlled variable data
        mdata = (double *)malloc(sizeof(double) * p); //Manipulated variable data
        //fxdata = (double *)malloc(sizeof(double) * nt);  //filtered xdata
        //fmdata = (double *)malloc(sizeof(double) * nt);
        xdatamod = (double *)malloc(sizeof(double) * p); //xdata from model
        a = (double **)malloc(sizeof(double *) * m);
        x = (double **)malloc(sizeof(double *) * m);
        xmodel = (double **)malloc(sizeof(double *) * m);
        paramtrs = (double **)malloc(sizeof(double *) * n);
        for(i=0; i<m; i++)
        {
                a[i] = (double *)malloc(sizeof(double)*n);
                x[i] = (double *)malloc(sizeof(double)*1);
                xmodel[i] = (double *)malloc(sizeof(double)*1);
        }
```

```
  for(i=0; i<n; i++)
  {
          paramtrs[i] = (double *)malloc(sizeof(double)*1);
  }
  //fp1=fopen("xdata1.txt","r");  //Data for control variable x(output).
  fp1=fopen("outs1.txt","r");
  //for(i=0; i<p; i++)
  for(i=0; i<110; i++)
  {
         fscanf(fp1,"%lf",&xdata[i]);
         //printf("%lf\n",xdata[i]);
  }
  fp2=fopen("mdata1.txt","r");  //Data for manipulated variable m (input).
  for(i=0; i<p; i++)
  {
         fscanf(fp2,"%lf",&mdata[i]);
         //printf("%lf\n",mdata[i]);
  }



 init_matrix(a,m,n);
 for(i=0; i<m; i++) //Feed Column by Column to form a matrix A.
 {
         for(j=0; j<n; j++)
         {
                 if(j<o)
                 {
                         k=(i-j)+o/2+(ct-nt+1);
                         a[i][j]=xdata[k];
                 }
                 else
                 {
                         k=(i-j+o)+o/2+(ct-nt+1);
                         a[i][j]=mdata[k];
                 }
         }
 }
//print_matrix(a,m,n);

//Enter column matrix X
for(i=0; i<m; i++)
{
        for(j=0; j<1; j++)
        {
                k=(i-j+1)+o/2+(ct-nt+1);
                x[i][j]=xdata[k];
        }
}
//print_matrix(x,m,1);
paramtrs=least_square(a,m,n,x);
```

```
    //printf("Solution Vector(Matrix):\n");
    //print_matrix(paramtrs,n,1);
    xmodel=matrix_mult(a,m,n,paramtrs,n,1): //x values from model.
    //print_matrix(xmodel,m,1);
    //printf("x[%d]=%lf",ct,xmodel[m-1][0]):

    /*Printing the output from the model*/
    for(i=0; i<nt; i++)
    {
        for(j=0; j<1; j++)
        {
            if(i<(o+ct-nt+1))
            {
                xdatamod[i]=xdata[i]: //x values from model and actual process.
            }
            else
            {
                xdatamod[i]=xmodel[i-o-(ct-nt+1)][j];
            }
        }
    }
    /*Prediction for next NP values from the current value with the model*/

    /*Prediction for next np values. */
    for(i=0; i<np; i++)
    {
        sum=0.0;
        for(j=0; j<o; j++)
        {
            sum=sum+(paramtrs[j][0])*xdatamod[ct+i-
j]+(paramtrs[j+o][0])*mdata[ct+i-j];
            //printf("%lf\t",sum);
        }
        //printf("\n");
        xdatamod[ct+i+1]=sum;
        printf("%lf\t %lf\n",xdatamod[ct+i+1], xdata[ct+i+1]);
    }


    fp3=fopen("xdat1mod.txt","w");

    for(i=0; i<(nt+np); i++)
    {
        fprintf(fp3,"%lf\n",xdatamod[i]);
    }


    return;

}
```

```
double **least_square(double **a, int m, int n, double **x)
{
        int i,j;
        double  **ata, **atainvat, **at, **theta;
        theta = (double **)malloc(sizeof(double *) * n); /* Column matrix of the parameters,
a1,a2,....,b1,b2, ...*/
        at = (double **)malloc(sizeof(double *) * n);
        ata = (double **)malloc(sizeof(double *) * n);
        atainvat = (double **)malloc(sizeof(double *) * n);
        for(i=0;i<n;i++)
        {
                atainvat[i] = (double *)malloc(sizeof(double)*m);
                at[i] = (double *)malloc(sizeof(double)*m);
                ata[i] = (double *)malloc(sizeof(double)*n);
                theta[i] = (double *)malloc(sizeof(double)*1);

        }

        at=matrix_tranps(a,m,n);  //transpose of matrix a
        ata=matrix_mult(at,n,m,a,m,n); //product of transpose of a and a
        init_matrix(atainvat,n,m);   //initialize the matrix
        atainvat=ainv_b(ata,at,n,m); //inverse of product of ata and at
        theta=matrix_mult(atainvat,n,m,x,m,1);  //Column matrix of the coefficients
        return theta;
}


double **ainv_b(double **a, double **matb, int n, int m)
{
        int i,j;
        double *b, **ainvb, **lo;
        b = (double *)malloc(sizeof(double) * n);
        ainvb = (double **)malloc(sizeof(double *) * n);
        lo = (double **)malloc(sizeof(double *) * n);
        for(i=0;i<n;i++)
        {
                ainvb[i] = (double *)malloc(sizeof(double)*m);
                lo[i] = (double *)malloc(sizeof(double)*n);
        }
        init_matrix(ainvb,n,m);
        unit_matrix(lo,n);
        ludcmp(a, lo, n);

        for(j=0; j<m; j++)
        {
                for(i=0; i<n; i++)
                {
                        b[i]=matb[i][j];

                }
                        luforwdsb(lo, b, n);
```

```
                        lubksb(a, b, n);
                        for(i=0; i<n; i++)
                        {
                                ainvb[i][j]=b[i];
                        }

        }
        return ainvb;
}
void init_matrix(double **a,int n, int m)
{
        int i,j;

        for(i=0;i<n;i++)
        {
                for(j=0;j<m;j++)
                {
                        a[i][j] = 0;
                }
        }
}
void ludcmp(double **a, double **lo, int n)
/*This subroutine decomposes a given square matrix of dimension n to its
upper triangular matrix a and lower triangular matrix lo. This routine fails
when the pivot element happen to be zero or near zero.*/
{

        int i,j,k;
        double mfactor;

        for(k=1;k<n;k++)
        {
                for(i=k;i<n;i++)
                {
                        if(a[k-1][k-1]==0)
                        {
                                a[k-1][k-1]=0.00000001;
                        }
                        mfactor=a[i][k-1]/a[k-1][k-1];
                        lo[i][k-1]=mfactor;
                        for(j=k-1;j<n;j++)
                        {
                            a[i][j]=a[i][j]-mfactor*a[k-1][j];
                        }

                }
        }


}
```

```
void luforwdsb(double **lo, double *b, int n)
{
        /*Forward sweep*/
        int i,j;
        double sum;
        for(i=1; i<n; i++)
        {
                sum=0.0;
                for(j=0;j<i;j++)
                {
                        sum=sum+lo[i][j]*b[j];
                }
                b[i]=b[i]-sum;
        }
}
void lubksb(double **a, double *b, int n)
{
        int i,j;
        double sum;
        /*Backward sweep*/
        /*The solution comes as a 'b' vector. If you want as x' vector, you
        can simply put x in the left hand side and define at the beginning.*/
        b[n-1]=b[n-1]/a[n-1][n-1]; /*The last element*/
        for(i=n-2;i>=0;i--)
        {
                sum=0.0;
                for(j=i+1;j<n;j++)
                {
                        sum=sum+a[i][j]*b[j];
                }
                b[i]=(b[i]-sum)/a[i][i];
        }


}

double **get_matrix(double **a,int n, int m)
{
        int i,j;
        a = (double **)malloc(sizeof(double *) * n);
        for(i=0;i<n;i++)
        {
                a[i] = (double *)malloc(sizeof(double)*m);
        }

        for(i=0;i<n;i++)
        {
                for(j=0;j<m;j++)
                {
                        printf("\n Enter element mat[%d][%d]:",i,j);
```

```
                        fflush(stdin);
                        scanf("%lf",&a[i][j]);
                }
        }

        return a;
}
double *get_vector(double *b,int n)
{
        int i;
        b=(double *)malloc(n*sizeof(double));
        for(i=0;i<n;i++)
        {
                printf("\nENTER THE VALUES OF b[%d]:",i);
                fflush(stdin);
                scanf("%lf",&b[i]);
        }
        return b;
}


void print_matrix(double **a,int n, int m)
{
        int i,j;
        for(i=0;i<n;i++)
        {
                for(j=0;j<m;j++)
                {
                        printf("%lf\t",a[i][j]);
                }
                printf("\n");
        }
}

void unit_matrix(double **lo, int n)
{
        int i,j;

        for(i=0; i<n; i++)
        {
                for(j=0; j<n; j++)
                {
                        if(i==j)
                        {
                                lo[i][j]=1.0;
                        }
                        else
                        {
                                lo[i][j]=0.0;
                        }
                }
```

```
		}
	}
	double **matrix_mult(double **a, int r1,int c1,double **b,int r2,int c2)
	{
		int i,j,k;
		double **c;
		if(c1 != r2)
		{
			return NULL;
		}

		c = (double **)malloc(sizeof(double *)*r1);
		for(i=0;i<r1;i++)
		{
			c[i] = (double *)malloc(sizeof(double)*c2);
		}

		for(i=0;i<r1;i++)
		{
			for(j=0;j<c2;j++)
			{
				c[i][j] = 0;
				for(k = 0;k<r2;k++)
				{
					c[i][j] += a[i][k] * b[k][j];
				}
			}
		}

		return c;

	}
	double **matrix_tranps(double **a, int nrows,int ncols)
	{
		int i,j;
		double **at;
		at= (double **)malloc(sizeof(double *)*ncols);
		for(i=0;i<ncols;i++)
		{
			at[i] = (double *)malloc(sizeof(double)*nrows);
		}

		for(i=0; i<ncols; i++)
		{
			for(j=0; j<nrows; j++)
			{
				at[i][j]=a[j][i];
			}
		}
		return at;
	}
```

5. Source code in C for Wavelet-based identification. Program name: *WLSI.C*

```
/*Sri Guru Gauranga Jayatah*/
/*Process identification using Wavelet-based method.*/
#include<stdio.h>
#include<math.h>

double **get_matrix(double **,int n, int m);
double *get_vector(double *,int n);
void init_matrix(double **,int n, int m);
void unit_matrix(double **, int n);
void print_matrix(double **,int n, int m);
void ludcmp(double **, double **, int n);
void luforwdsb(double **, double *, int n);
void lubksb(double **, double *, int n);
double  **ainv_b(double **a, double **matb, int n, int m);
double **matrix_mult(double **a, int r1,int c1,double **b,int r2,int c2);
double **matrix_tranps(double **a, int nrows,int ncols);
double **least_square(double **a, int m, int n, double **x);

FILE *fp1,*fp2, *fp3;
main()
{
        int i,j,k,m,n,o,p,s,ct,nt,np,fc;
        double **a, **x, **paramtrs,*xdata, *mdata, **xmodel, *xdatamod, sum;
        double **w, **wa, **wx;

        /*Here A is the matrix of output and manipulated variables.
        X is a vector of output variables. Here it is taken as a column
        matrix for ease of matrix multiplication. Similarly PARAMTRS is also
        a vector of parameters though it is taken as a matrix.*/
        clrscr();
        p=900; //Total number of data
        nt=80; //  truncation horizon
        o=1;    //order of model
        s=0; //Scale of filteration
        np=10;  //length of prediction horizon
        ct=nt-1;//current time instant, ct ( should be >=nt-1)
        m=nt-o+1;  /* m and n are the dimensions of matrix A for least sqr.*/
        n=2*o;

        xdata = (double *)malloc(sizeof(double) * p); //Controlled variable data
        mdata = (double *)malloc(sizeof(double) * p); //Manipulated variable data
        //fxdata = (double *)malloc(sizeof(double) * nt);  //filtered xdata
        //fmdata = (double *)malloc(sizeof(double) * nt);
        xdatamod = (double *)malloc(sizeof(double) * p); //xdata from model
        a = (double **)malloc(sizeof(double *) * m);
        x = (double **)malloc(sizeof(double *) * m);
        xmodel = (double **)malloc(sizeof(double *) * m);
        paramtrs = (double **)malloc(sizeof(double *) * n);
```

```
w=(double **)malloc(sizeof(double *)*n); //wavelet matrix
wa=(double **)malloc(sizeof(double *)*n);
wx=(double **)malloc(sizeof(double *)*n);
for(i=0; i<m; i++)
{
        a[i] = (double *)malloc(sizeof(double)*n);
        x[i] = (double *)malloc(sizeof(double)*1);
        xmodel[i] = (double *)malloc(sizeof(double)*1);
}
for(i=0; i<n; i++)
{
        paramtrs[i] = (double *)malloc(sizeof(double)*1);
        w[i]=(double *)malloc(sizeof(double)*m);
        wa[i]=(double *)malloc(sizeof(double)*n);
        wx[i]=(double *)malloc(sizeof(double)*1);

}
//fp1=fopen("xdata1.txt","r"); //Data for control variable x(output).
fp1=fopen("outs1.txt","r");
//for(i=0; i<p; i++)
for(i=0; i<110; i++)
{
        fscanf(fp1,"%lf",&xdata[i]);
        //printf("%lf\n",xdata[i]);
}
fp2=fopen("mdata1.txt","r"); //Data for manipulated variable m (input).
for(i=0; i<p; i++)
{
        fscanf(fp2,"%lf",&mdata[i]);
        //printf("%lf\n",mdata[i]);
}



init_matrix(a,m,n);
for(i=0; i<m; i++) //Feed Column by Column to form a matrix A.
{
        for(j=0; j<n; j++)
        {
                if(j<o)
                {
                        k=(i-j)+o/2+(ct-nt+1);
                        a[i][j]=xdata[k];
                }
                else
                {
                        k=(i-j+o)+o/2+(ct-nt+1);
                        a[i][j]=mdata[k];
                }
        }
}
```

```
//print_matrix(a,m,n);

//Enter column matrix X
for(i=0; i<m; i++)
{
        for(j=0; j<1; j++)
        {
                k=(i-j+1)+o/2+(ct-nt+1);
                x[i][j]=xdata[k];
        }
}
/*Forming the Haar wavelet Matrix, W*/
for(i=0; i<n; i++)
{
        for(j=0; j<m; j++)
        {
                if(j>=pow(2,s)*i && j<pow(2,s)*(k+1))
                {
                        w[i][j]=pow(2,-s/2);
                }
                else
                {
                        w[i][j]=0;
                }
        }
}
wa=matrix_mult(w,n,m,a,m,n);
wx=matrix_mult(w,n,m,x,m,1);
//print_matrix(x,m,1);
paramtrs=least_square(wa,n,n,wx);
//printf("Solution Vector(Matrix):\n");
//print_matrix(paramtrs,n,1);
xmodel=matrix_mult(a,m,n,paramtrs,n,1); //x values from model.
//print_matrix(xmodel,m,1);
//printf("x[%d]=%lf",ct,xmodel[m-1][0]);

/*Printing the output from the model*/
for(i=0; i<nt; i++)
{
        for(j=0; j<1; j++)
        {
                if(i<(o+ct-nt+1))
                {
                        xdatamod[i]=xdata[i]; //x values from model and actual process.
                }
                else
                {
                        xdatamod[i]=xmodel[i-o-(ct-nt+1)][j];
                }
        }
}
```

```
	/*Prediction for next NP values from the current value with the model*/

	/*Prediction for next np values. */
	for(i=0; i<np; i++)
	{
		sum=0.0;
		for(j=0; j<o; j++)
		{
			sum=sum+(paramtrs[j][0])*xdatamod[ct+i-
j]+(paramtrs[j+o][0])*mdata[ct+i-j];
			//printf("%lf\t",sum);
		}
		//printf("\n");
		xdatamod[ct+i+1]=sum;
		printf("%lf\t %lf\n",xdatamod[ct+i+1], xdata[ct+i+1]);
	}


	fp3=fopen("xdat1mod.txt","w");

	for(i=0; i<(nt+np); i++)
	{
		fprintf(fp3,"%lf\n",xdatamod[i]);
	}


	return;

}

double **least_square(double **a, int m, int n, double **x)
{
	int i,j;
	double  **ata, **atainvat, **at, **theta;
	theta = (double **)malloc(sizeof(double *) * n); /*Prameters, a1,a2,...,b1,b2, ...*/
	at = (double **)malloc(sizeof(double *) * n);
	ata = (double **)malloc(sizeof(double *) * n);
	atainvat = (double **)malloc(sizeof(double *) * n);
	for(i=0;i<n;i++)
	{
		atainvat[i] = (double *)malloc(sizeof(double)*m);
		at[i] = (double *)malloc(sizeof(double)*m);
		ata[i] = (double *)malloc(sizeof(double)*n);
		theta[i] = (double *)malloc(sizeof(double)*1);

	}

	at=matrix_tranps(a,m,n);  //transpose of matrix a
	ata=matrix_mult(at,n,m,a,m,n); //product of transpose of a and a
	init_matrix(atainvat,n,m);  //initialize the matrix
	atainvat=ainv_b(ata,at,n,m); //inverse of product of ata and at
```

```
        theta=matrix_mult(atainvat,n,m,x,m,1);  //Column matrix of the coefficients
        return theta;
}

double **ainv_b(double **a, double **matb, int n, int m)
{
        int i,j;
        double *b, **ainvb, **lo;
        b = (double *)malloc(sizeof(double) * n);
        ainvb = (double **)malloc(sizeof(double *) * n);
        lo = (double **)malloc(sizeof(double *) * n);
        for(i=0;i<n;i++)
        {
                ainvb[i] = (double *)malloc(sizeof(double)*m);
                lo[i] = (double *)malloc(sizeof(double)*n);
        }
        init_matrix(ainvb,n,m);
        unit_matrix(lo,n);
        ludcmp(a, lo, n);

        for(j=0; j<m; j++)
        {
                for(i=0; i<n; i++)
                {
                        b[i]=matb[i][j];

                }
                        luforwdsb(lo, b, n);
                        lubksb(a, b, n);
                        for(i=0; i<n; i++)
                        {
                                ainvb[i][j]=b[i];
                        }

        }
        return ainvb;
}
void init_matrix(double **a,int n, int m)
{
        int i,j;

        for(i=0;i<n;i++)
        {
                for(j=0;j<m;j++)
                {
                        a[i][j] = 0;
                }
        }
}
void ludcmp(double **a, double **lo, int n)
/*This subroutine decomposes a given square matrix of dimension n to its
```

```
upper triangular matrix a and lower triangular matrix lo. This routine fails
when the pivot element happen to be zero or near zero.*/
{

        int i,j,k;
        double mfactor;

        for(k=1;k<n;k++)
        {
                for(i=k;i<n;i++)
                {
                        if(a[k-1][k-1]==0)
                        {
                                a[k-1][k-1]=0.00000001;
                        }
                        mfactor=a[i][k-1]/a[k-1][k-1];
                        lo[i][k-1]=mfactor;
                        for(j=k-1;j<n;j++)
                        {
                            a[i][j]=a[i][j]-mfactor*a[k-1][j];
                        }

                }
        }


}


void luforwdsb(double **lo, double *b, int n)
{
        /*Forward sweep*/
        int i,j;
        double sum;
        for(i=1; i<n; i++)
        {
                sum=0.0;
                for(j=0;j<i;j++)
                {
                        sum=sum+lo[i][j]*b[j];
                }
                b[i]=b[i]-sum;
        }
}
void lubksb(double **a, double *b, int n)
{
        int i,j;
        double sum;
        /*Backward sweep*/
        /*The solution comes as a 'b' vector. If you want as x' vector, you
        can simply put x in the left hand side and define at the beginning.*/
```

```
        b[n-1]=b[n-1]/a[n-1][n-1]; /*The last element*/
        for(i=n-2;i>=0;i--)
        {
                sum=0.0;
                for(j=i+1;j<n;j++)
                {
                        sum=sum+a[i][j]*b[j];
                }
                b[i]=(b[i]-sum)/a[i][i];
        }


}
double **get_matrix(double **a,int n, int m)
{
        int i,j;
        a = (double **)malloc(sizeof(double *) * n);
        for(i=0;i<n;i++)
        {
                a[i] = (double *)malloc(sizeof(double)*m);
        }

        for(i=0;i<n;i++)
        {
                for(j=0;j<m;j++)
                {
                        printf("\n Enter element mat[%d][%d]:",i,j);
                        fflush(stdin);
                        scanf("%lf",&a[i][j]);
                }
        }

        return a;
}
double *get_vector(double *b,int n)
{
        int i;
        b=(double *)malloc(n*sizeof(double));
        for(i=0;i<n;i++)
        {
                printf("\nENTER THE VALUES OF b[%d]:",i);
                fflush(stdin);
                scanf("%lf",&b[i]);
        }
        return b;
}

void print_matrix(double **a,int n, int m)
{
        int i,j;
        for(i=0;i<n;i++)
```

```
        {
                for(j=0;j<m;j++)
                {
                        printf("%lf\t",a[i][j]);
                }
                printf("\n");
        }
}
void unit_matrix(double **lo. int n)
{
        int i,j;

        for(i=0; i<n; i++)
        {
                for(j=0; j<n; j++)
                {
                        if(i==j)
                        {
                                lo[i][j]=1.0;
                        }
                        else
                        {
                                lo[i][j]=0.0;
                        }
                }
        }
}
double **matrix_mult(double **a, int r1,int c1,double **b,int r2,int c2)
{
        int i,j,k;
        double **c;
        if(c1 != r2)
        {
                return NULL;
        }
        c = (double **)malloc(sizeof(double *)*r1);
        for(i=0;i<r1;i++)
        {
                c[i] = (double *)malloc(sizeof(double)*c2);
        }
        for(i=0;i<r1;i++)
        {
                for(j=0;j<c2;j++)
                {
                        c[i][j] = 0;
                        for(k = 0;k<r2;k++)
                        {
                                c[i][j] += a[i][k] * b[k][j];
                        }
                }
        }
```

```
        return c;
}
double **matrix_tranps(double **a, int nrows,int ncols)
{
        int i,j;
        double **at;
        at= (double **)malloc(sizeof(double *)*ncols);
        for(i=0;i<ncols;i++)
        {
                at[i] = (double *)malloc(sizeof(double)*nrows);
        }

        for(i=0; i<ncols; i++)
        {
                for(j=0; j<nrows; j++)
                {
                        at[i][j]=a[j][i];
                }
        }
        return at;
}
```

6. Source code in C for manual control. Program name: *MANCON.C*

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<conio.h>
#include<dos.h>
#include<time.h>
#define BASE 0x218
FILE *fp;
time_t init,final;
void steady_state(float v0);
void send_output(float v0);
float read_temp();
void main()
{
        int i,j;
        float v0, a_time=0.0, temp;
        clrscr();
        fp=fopen("c:\\tc3\\bin\\mancon20.txt","a+");
        fprintf(fp,"HARE KRISHNA\n");
        fprintf(fp,"SS FOR DMCx.TXT\n");
        fprintf(fp, "DATE : 25 July 2004\n");
        //fprintf(fp,"Finding the step response for DMC\n");
        //fprintf(fp,"Finding the transfer function for the process.\n");
        fprintf(fp,"Sampling Time = 1 second\n");
```

```
        printf("Enter the initial controller position: ");
        scanf("%f",&v0);
        fprintf(fp,"Initial controller position=%f\n",v0);
        printf("HARE KRISHNA !\n");
        printf("<HIT ANY KEY TO START>\n");
        steady_state(v0);
        if(!fp)  exit(-1);
        while(!kbhit())
        {
         printf("WAIT - SYSTEM BEING BROUGHT TO STEADY STATE\n\n");
         printf("Check connections before hitting any key\n\n");
         printf("Hit any key when steady state is reached\n\n\n");
         delay(2000);
         temp = read_temp();
         printf("TEMPERATURE = %f\n", temp);
        }
        printf("Enter the final controller position: ");
        scanf("%f", &v0);
        fprintf(fp,"Final controller position=%f\n",v0);
        fprintf(fp,"Temperature(deg. C)\n");
                while(!kbhit())
                {
                        init = time(NULL);
                        send_output(v0);
                        delay(1000);
                        temp = read_temp();
                        final = time(NULL);
                        a_time += difftime(final,init);
                        printf("\tTemp=%f\t Time=%f\n",temp, a_time);
                        fprintf(fp,"%f\n",temp);
                }
        fclose(fp);
}
void steady_state(float v0)
{
        int n,lb,hb;
        while(!kbhit())
        {
                n = (int) ((v0+5.0)*4095.0/10);
                lb = n & 0xff;
                hb = (int) (n-lb)/256.0;
                outportb (BASE + 2, lb);
                outportb (BASE + 3, hb);
        }
        getch();
        clrscr();
}
void send_output(float v0)
{
        int n,lb,hb;
        n = (int) ((v0 + 5.0)*4095.0/10.0);
```

```
        lb = n & 0xff;
        hb = (int) (n - lb)/256.0;
        outportb (BASE + 2, lb);
        outportb (BASE + 3, hb);
}
float read_temp()
{
        int i,lb,hb,status,adc_no;
        float temp,voltage=0.0;
        for(i=0; i<2000; i++)
        {
                outportb (BASE+0, 0);
                outportb (BASE+1, 0);
                status = inport (BASE+0);
                while(status < 128)
                        status = inport (BASE+0);
                lb = inportb (BASE+2);
                hb = inportb (BASE+3);
                hb = hb & 0x0f;
                adc_no = lb + 256*hb;
                voltage += (adc_no/4096.0)*10.0 - 5.0;
        }
        temp = (voltage/2000.0)*100.0;
       voltage = voltage/2000.0;
        return temp;

}
```

7. Source code in C for PID control of temperature. Program name: *PID.C*

```
#include<stdio.h>
#include<math.h>
#include<stdlib.h>
#include<conio.h>
#include<dos.h>
#include<time.h>
#define BADR 0x218
float read_temp();
void send_output(float);
FILE *fp;
clock_t start,end;
time_t init,final;
void main()
{
  float kc,ki,kd,taui,taud,s_time,tot_time,set_pt;
  float cwitemp, sscwot, cwfr, hwitemp, sshwfr;
  float sum=0.0,prev_error = 0.0,prev_del_error = 0.0;
  float el_t=0.0, init_co,temp,error,del_error,del_c,co,ts, volt;
  clrscr();
  kc=100.0;           /* proportiona gain */
  taui=4.0;             /* Integral time in seconds */
```

```
 taud=1.0;             /* Derivative time in seconds */
 s_time=1.0;           /* Sampling time in seconds */
 tot_time=300.0;       /* Total time in seconds */
 set_pt=45.0;          /* Set point in degree C */
 cwitemp=31.6;         /* Cold water inlet temp in degree C */
 hwitemp=70.0;         /* Hot water inlet temp in degree C, +_3 */
 sscwot=35.2;         /* Steady state cold water outlet temp in degree C */
 cwfr=200.0;           /* Cold water flow rate in cc/min  */
 sshwfr=10.0; /* Steady state hot water flow rate in cc/min  */
 fp=fopen("c:\\tcplus\\bin\\pidtc7.txt","w");
 fprintf(fp, "DATE: 3 JUNE 2004\n");
 fprintf(fp,"PID control of the heat exchanger with co converted to Voltage\n");
 fprintf(fp,"Set point=%f\n",set_pt);
 fprintf(fp,"Steady state hot water flow rate =%f\n",sshwfr);
 fprintf(fp,"Cold water inlet temperature =%f\n",cwitemp);
 fprintf(fp,"Steady state cold water outlet temperature =%f\n",sscwot);
 fprintf(fp,"Cold water flow rate =%f\n",cwfr);
 fprintf(fp,"Hot water inlet temp =%f\n",hwitemp);
 fprintf(fp,"Proportional gain =%f\n",kc);
 fprintf(fp,"Integral time (seconds)=%f\n",taui);
 fprintf(fp,"Derivative time (seconds) =%f\n",taud);
 fprintf(fp,"Sampling time (seconds)=%f\n",s_time);
 fprintf(fp,"Total time (seconds)=%f\n",tot_time);
 fprintf(fp,"\nTime(sec)\tController Output\tC.W. Outlet Temp(degree C)\n");
 printf("Enter Initial Value of Controller Output, CO: \n");
 scanf("%f",&init_co);

 while(!kbhit())
 {
  temp=read_temp();
  printf("Temperature=%f\n\n",temp);
  printf("System is being brought to steady state\n\n");
  printf("Hit any key when steady state is reached\n\n\n");
  delay(5000);
 }
 while( el_t <= tot_time)
 {
  start=clock();
  init=time(NULL);
  temp=read_temp();
  error = set_pt-temp;
  del_error = error - prev_error;
  prev_del_error = del_error - prev_del_error;
  ki = kc/taui;
  kd = kc*taud;
  del_c=(kc*del_error+((s_time*ki)*error)+((kd/s_time)*(del_error-prev_del_error)));
  sum = sum + del_c;
  co = init_co + sum;
  if(co > 4.0)
  co = 4.0;
  else if(co < 0.0)
```

```
    co = 0.5;
    send_output(co);
    end = clock();
    ts = (end-start)/CLK_TCK; /*Time spent in seconds  up to sixth decimal places*/
    if(ts<s_time)
    {
     ts =s_time-ts;
     delay((int)(ts*1000)); /* Delay in millisecond */
    }
    final=time(NULL);
    el_t += difftime(final,init); /* Time lapsed as an integer count */
    printf("Error=%f\t Time=%f\n",error,el_t);
    fprintf(fp,"%f\t %f\t\t %f\n",el_t,co,temp);
    prev_error =error;
    prev_del_error = del_error;
   }
fclose(fp);
}

float read_temp()
{
  int i,n,lb,hb,status,adc_no;
  float voltage = 0.0, temp;
  for(i=0; i<500; i++)
  {
    outportb (BADR+0, 0);
    outportb (BADR+1, 0);
    status = inport (BADR+0);
    while(status < 128)
    status = inport (BADR+0);
    lb = inportb (BADR+2);
    hb = inportb (BADR+3);
    hb = hb & 0x0f;
    adc_no = lb + 256*hb;
    voltage += (adc_no/4096.0)*10.0 - 5.0;
  }
    temp = (voltage/500.0)*100.0 + 1.18;
    return temp;
}

void send_output(float co)
{
  int n,lb,hb;
  n = (int)(co + 5.0)*4095.0/10.0;
  lb = n & 0xff;
  hb = (int)(n-lb)/256.0;
  outportb(BADR + 2, lb);
  outportb(BADR + 3, hb);
}
```

8. Source code in C for dynamic matrix control of temperature. Program name: *DMC.C*

```
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<math.h>
#include<dos.h>
#include<time.h>

#define BASE 0x218  /* set base address */

void init_matrix(double **,int n, int m);
void unit_matrix(double **, int n);
void weighting_ matrix(double **, int n, double msp);
void dynamic_matrix(double **a, int np, int nc, double *beta);
void print_matrix(double **,int n, int m);
void ludcmp(double **a, double **lo, int n);
void luforwdsb(double **lo, double *b, int n);
void lubksb(double **a, double *b, int n);
double **matrix_trps(double **a, int nrows,int ncols);
double **matrix_add(double **a, int r1,int c1,double **b,int r2,int c2);
void matrix_mult(double **a,int r1,int c1,double **b,int r2,int c2, double **c);
double **ainv_b(double **a, double **matb, int n);
void steady_state(double v0);
void send_output(double v0);
double read_temp();
FILE *fp, *fp1;
clock_t start,end;
time_t init,final;

main()
{
        double **deltam,**deltam_old,**a,**at,**ata,**wtmat,**inputmat;
        double **perrmat,**sysmat,*xol,*beta;
        double temp,set_pt, sum,ts, msp, c_op, init_c_op, el_t=0.0,error;
        double cwitemp,hwitemp,cwfr,sum1=0.0,s_time,tot_time,ise=0.0;
        int i,j,k,np,nc,nt;
        clrscr();
        nt=150; /* Truncation Horizon, which covers 90 to 95% of the steady state response */
        np=32;  /*Projection horizon */
        nc=16;          /* Control Horizon, typically set at 50% of NP */
        s_time=1.0;     /* Sampling time in seconds */
        tot_time=301.0; /* Time of operation */
        msp =200.0;     /*Move supression parameter, should be a positive real number */

        cwitemp=16.7;  /* Cold water inlet temp in degree C */
        hwitemp=70.0;  /* Hot water inlet temp in degree C, +_4.0 */
        cwfr=200.0;     /* Cold water flow rate in cc/min  */

        fp=fopen("c:\\tc3\\bin\\dmc1nt3.txt","a+");
        fprintf(fp,"HARE KRISHNA\n");
```

```
fprintf(fp, "DATE: 27 JAN 2005\n");
fprintf(fp,"Cold water inlet temperature =%lf\n",cwitemp);
fprintf(fp,"Cold water flow rate =%lf\n",cwfr);
//fprintf(fp,"Cold water flow rate (initial)=%lf\n",cwfr);
//fprintf(fp,"Cold water flow rate (final)= \n");
fprintf(fp,"Hot water inlet temp =%lf\n",hwitemp);
fprintf(fp,"Truncation Horizon, NT =%d\n",nt);
fprintf(fp,"Projection Horizon, NP =%d\n",np);
fprintf(fp,"Control Horizon, NC=%d\n",nc);
fprintf(fp,"Move supression parameter, msp=%lf\n",msp);
fprintf(fp,"Sampling time (seconds)=%lf\n",s_time);
fprintf(fp,"Total time (seconds)=%lf\n",tot_time);

beta =(double *)malloc((nt+np)*sizeof(double));
xol =(double *)malloc(np*sizeof(double));

a = (double **)malloc(sizeof(double *) * np);
at = (double **)malloc(sizeof(double *) * nc);
ata = (double **)malloc(sizeof(double *) * nc);
wtmat = (double **)malloc(sizeof(double *) * nc);
inputmat = (double **)malloc(sizeof(double *) * nc);
sysmat = (double **)malloc(sizeof(double *) * nc);
deltam = (double **)malloc(sizeof(double *) * nc);
deltam_old = (double **)malloc(sizeof(double *) * nt);
perrmat = (double **)malloc(sizeof(double *) * np);

for(i=0; i<np; i++)
{
        a[i] = (double *)malloc(sizeof(double)*nc);
        perrmat[i] = (double *)malloc(sizeof(double)*1);
}
for(i=0; i<nt; i++)
{
    deltam_old[i] = (double *)malloc(sizeof(double)*1);
}

for(i=0;i<nc;i++)
{

        at[i] = (double *)malloc(sizeof(double)*np);
        ata[i] = (double *)malloc(sizeof(double)*nc);
        wtmat[i] = (double *)malloc(sizeof(double)*nc);
        inputmat[i] = (double *)malloc(sizeof(double)*nc);
        sysmat[i] = (double *)malloc(sizeof(double)*np);
        deltam[i] = (double *)malloc(sizeof(double)*1);

}

fp1 = fopen("c:\\tc3\\bin\\dmcstep1.txt","r");
fflush(stdin);
for(i=0; i<(nt+np); i++)
```

```
  {
   fscanf(fp1,"%lf", &beta[i]);
  }
 printf("Enter the initial controller output: ");
 scanf("%lf",&init_c_op);
 printf("HARE KRISHNA !\n");
 printf("<HIT ANY KEY TO START>\n");
 fprintf(fp,"Initial controller output=%f\n",init_c_op);
 steady_state(init_c_op);

 while(!kbhit())
 {
        printf("WAIT - SYSTEM BEING BROUGHT TO STEAD STATE\n\n");
        printf("Check connections before hitting any key\n");
        printf("Hit any key when steady state is reached\n\n\n");
        temp = read_temp();
        printf("Temperature=%f\n\n\n",temp);
        delay(1000);
 }

 printf("Enter Second initial controller output: ");
 scanf("%lf",&init_c_op);
 printf("HARE KRISHNA !\n");
 printf("<HIT ANY KEY TO START>\n");
 fprintf(fp,"Initial controller output=%f\n",init_c_op);
 steady_state(init_c_op);

 while(!kbhit())
 {
        printf("WAIT-SYSTEM BEING BROUGHT TO STEADY STATE\n\n");
        printf("Check connections before hitting any key\n");
        printf("Hit any key when steady state is reached\n\n\n");
        temp = read_temp();
        printf("Temperature=%f\n\n\n",temp);
        delay(1000);
 }

 printf("Enter the set point in cold water outlet temperature:");
 scanf("%lf",&set_pt);
 fprintf(fp,"Steady state cold water outlet temperature =%lf\n",temp);
 fprintf(fp,"Set point=%lf\n",set_pt);
 fprintf(fp,"\n\n\tC_OP\t\t\tTemp(C)\t\tTime spent(sec)\n");
 /*Steady state values*/
 /*The current instant is 0th instant*/
 init_matrix(deltam_old,nt,1);
 for(k=0; k<nt; k++)
 {
        for(j=0; j<1; j++)
        {
          deltam_old[k][j]=0.0;  /* Control matrix, deltaM_old */
        }
```

```
}
init_matrix(perrmat,np,1);
for(i=0; i<np; i++)
{
  xol[i]=temp;          /* Open loop values of x, temp */
  for(j=0; j<1; j++)
  {
       perrmat[i][j]=set_pt-xol[i]; /*Pseudo-error Matrix*/
  }
}

init_matrix(a,np,nc);     /* Initialize the dynamic matrix A */
init_matrix(at,nc,np);
init_matrix(ata,nc,nc);
init_matrix(wtmat,nc,nc);
init_matrix(inputmat,nc,nc);
init_matrix(sysmat,nc,np);
init_matrix(deltam,nc,1);

dynamic_matrix(a,np,nc,beta);  /*Entering the values for dynamic matrix */
at = matrix_trps(a,np,nc);          /* Transpose of A */
matrix_mult(at,nc,np,a,np,nc,ata);          /* Transpose of A into A */

weighting_matrix(wtmat,nc,msp);  /* Weighting matrix with same msp value in all the
diagonal element */
inputmat = matrix_add(ata,nc,nc,wtmat,nc,nc);  /*Input matrix*/
sysmat = ainv_b(inputmat,at,nc);                       /*System Matrix*/


/*Calculating the future control moves, deltam[0] to deltam[NC-1]*/


while(el_t <= tot_time)
{
 init=time(NULL);
 start = clock();
 error = set_pt - temp;             /*Actual error*/
 ise = ise + pow(error,2)*s_time;    /*Integral of the square error*/
 /* Control action */
 matrix_mult(sysmat,nc,nc,perrmat,nc,1,deltam); /*Control matrix, deltaM*/
  /*Implement the first control move from the optimum future control
moves, that is deltam[0][0]*/

 sum1 += deltam[0][0];    /*Summation of all "change in control moves"*/
c_op = init_c_op + sum1;  /*Control action to be taken*/
   if(c_op > 4.0)
             c_op = 4.0;
   else if(c_op < 0.0)
             c_op = 0.0501021;
printf("C_OP=%lf\t",c_op);
send_output(c_op);
```

```
/*Shift the control moves and calculate the openloop and error
values*/
for(i=nt-1; i>=1; i--)
{
  for(j=0; j<1; j++)
   {
       deltam_old[i][j]=deltam_old[i-1][j];
   }
}
deltam_old[0][0]=deltam[0][0];     /* Shifting the control measures */
temp = read_temp();               /*On-line temp value*/


for(i=0;i<np;i++)
{
       sum=0.0;
       for(k=0; k<nt; k++)
       {
          sum=sum+(beta[i+1+k]-beta[1+k])*deltam_old[k][0];
       }
       xol[i] = temp + sum;             /*Open loop response*/
       perrmat[i][0]=set_pt-xol[i];    /*Pseudo-error matrix*/

}
end = clock();
       ts=(end-start)/CLK_TCK;
       if(ts<s_time)
         {
          delay((int)((s_time-ts)*1000));
         }
final=time(NULL);
el_t += difftime(final,init); /* Time lapsed as an integer count */
printf("Error=%lf\tTime=%lf\tTemp=%lf\n",error,el_t,temp);
fprintf(fp,"\t%lf\t\t %lf\t\t %lf\n",c_op,temp,ts);

} /*Main WHILE loop ends here*/
fprintf(fp,"\nISE=%lf\n",ise);
for(i=0;i<nc;i++)
{

       free(at[i]);
       free(ata[i]);
       free(wtmat[i]);
       free(inputmat[i]);
       free(sysmat[i]);
       free(deltam[i]);
}
free(at);
free(ata);
free(wtmat);
```

```
        free(inputmat);
        free(sysmat);
        free(deltam);
        for(i=0; i<np; i++)
        {
                free(a[i]);
                free(perrmat[i]);
        }
        for(i=0; i<nt; i++)
        {
            free(deltam_old[i]);
        }
        free(a);
        free(perrmat);
        free(deltam_old);

        fclose(fp);
        fclose(fp1);
        return 0;
}
void dynamic_matrix(double **a, int np, int nc, double *beta)
{
        int i,k;
        for(i=0; i<np; i++)
        {
                for(k=0; k<nc; k++)
                {
                        if(i<k)
                        {
                                a[i][k]=0.0;
                        }
                        else
                        {
                                a[i][k]=beta[i+1-k];
                        }
                }
        }
}

void init_matrix(double **a,int n,int m)
{
        int i,j;
        for(i=0;i<n;i++)
        {
                for(j=0;j<m;j++)
                {
                        a[i][j] = 0.0;
                }
        }
}
```

```
void print_matrix(double **a,int n, int m)
{
        int i,j;
        for(i=0;i<n;i++)
        {
                for(j=0;j<m;j++)
                {
                        printf("\t%lf",a[i][j]);
                }
                printf("\n");
        }
}

void matrix_mult(double **a, int r1,int c1,double **b,int r2,int c2, double **c)
{
        int i,j,k;
        if(c1 != r2)
        {
                c = NULL;
        }
        for(i=0;i<r1;i++)
        {
                for(j=0;j<c2;j++)
                {
                        c[i][j] = 0;
                        for(k = 0;k<r2;k++)
                        {
                                c[i][j] += a[i][k] * b[k][j];
                        }
                }
        }


}
double **matrix_add(double **a, int r1,int c1,double **b,int r2,int c2)
{
        int i,j,k;
        double **c;
        if(r1 != r2 || c1 != c2)
        {
                return NULL;
        }

        c = (double **)malloc(sizeof(double *)*r1);
        for(i=0;i<r1;i++)
        {
                c[i] = (double *)malloc(sizeof(double)*c2);
        }

        for(i=0;i<r1;i++)
        {
```

```
                for(j=0;j<c2;j++)
                {
                        c[i][j] = a[i][j] + b[i][j];

                }
        }

        return c;

}

double **ainv_b(double **a, double **matb, int n)
{
        int i,j;
        double *b, **ainvb, **lo;
        b = (double *)malloc(sizeof(double) * n);
        ainvb = (double **)malloc(sizeof(double *) * n);
        lo = (double **)malloc(sizeof(double *) * n);
        for(i=0;i<n;i++)
        {
                ainvb[i] = (double *)malloc(sizeof(double)*n);
                lo[i] = (double *)malloc(sizeof(double)*n);
        }
        init_matrix(ainvb,n,n);
        unit_matrix(lo,n);
        ludcmp(a, lo, n);

        for(j=0; j<n; j++)
        {
                for(i=0; i<n; i++)
                {
                        b[i]=matb[i][j];

                }
                        luforwdsb(lo, b, n);
                        lubksb(a, b, n);
                        for(i=0; i<n; i++)
                        {
                                ainvb[i][j]=b[i];
                        }

        }
        return ainvb;
}

void ludcmp(double **a, double **lo, int n)
/*This subroutine decomposes a given square matrix of dimension n to its
upper triangular matrix a and lower triangular matrix lo. This routine fails
when the pivot element happen to be zero or near zero. Initially lo is to be
defined as an identity matrix.*/
{
```

```
        int i,j,k;
        double mfactor;

        for(k=1;k<n;k++)
        {
                for(i=k;i<n;i++)
                {
                        mfactor=a[i][k-1]/a[k-1][k-1];
                        lo[i][k-1]=mfactor;
                        for(j=k-1;j<n;j++)
                        {
                            a[i][j]=a[i][j]-mfactor*a[k-1][j];
                        }

                }
        }


}


void luforwdsb(double **lo, double *b, int n)
{
        /*Forward sweep*/
        int i,j;
        double sum;
        for(i=1; i<n; i++)
        {
                sum=0.0;
                for(j=0;j<i;j++)
                {
                        sum=sum+lo[i][j]*b[j];
                }
                b[i]=b[i]-sum;
        }
}
void lubksb(double **a, double *b, int n)
{
        int i,j;
        double sum;
        /*Backward sweep*/
        /*The solution comes as a 'b' vector. If you want as x' vector, you
        can simply put x in the left hand side and define at the beginning.*/
        b[n-1]=b[n-1]/a[n-1][n-1]; /*The last element*/
        for(i=n-2;i>=0;i--)
        {
                sum=0.0;
                for(j=i+1;j<n;j++)
                {
                        sum=sum+a[i][j]*b[j];
```

```
				}
				b[i]=(b[i]-sum)/a[i][i];
			}


	}


	void unit_matrix(double **lo, int n)
	{
		int i,j;

		for(i=0; i<n; i++)
		{
			for(j=0; j<n; j++)
			{
				if(i==j)
				{
					lo[i][j]=1.0;
				}
				else
				{
					lo[i][j]=0.0;
				}
			}
		}
	}
	void weighting_matrix(double **lo, int n, double msp)
	{
		int i,j;

		for(i=0; i<n; i++)
		{
			for(j=0; j<n; j++)
			{
				if(i==j)
				{
					lo[i][j] = msp;
				}
				else
				{
					lo[i][j]=0.0;
				}
			}
		}
	}

	double **matrix_trps(double **a, int nrows,int ncols)
	{
		int i,j;
		double **at;
```

```
        at=(double **)malloc(sizeof(double *)*ncols);
        for(i=0; i<ncols; i++)
        {
                at[i]=(double *)malloc(sizeof(double)*nrows);
        }
        for(i=0; i<ncols; i++)
        {
                for(j=0; j<nrows; j++)
                {
                        at[i][j]=a[j][i];
                }
        }
        return at;
}
void steady_state(double v0)
{
        int n,lb,hb;
        while(!kbhit())
        { -
                n = (int) ((v0+5.0)*4095.0/10);
                lb = n & 0xff;
                hb = (int) (n-lb)/256.0;
                outportb (BASE + 2, lb);
                outportb (BASE + 3, hb);
        }
        getch();
        clrscr();
}
void send_output(double v0)
{
        int n,lb,hb;
        n = (int) ((v0 + 5.0)*4095.0/10.0);
        lb = n & 0xff;
        hb = (int) (n - lb)/256.0;
        outportb (BASE + 2, lb);
        outportb (BASE + 3, hb);
}

double read_temp()
{
        int i,lb,hb,status,adc_no;
        float temp,voltage=0.0;

        for(i=0; i<2000; i++)
        {
                outportb (BASE+0, 0);
                outportb (BASE+1, 0);
                status = inport (BASE+0);
                while(status < 128)
                        status = inport (BASE+0);
                lb = inportb (BASE+2);
```

```
                hb = inportb (BASE+3);
                hb = hb & 0x0f;
                adc_no = lb + 256*hb;
                voltage += (adc_no/4096.0)*10.0 - 5.0;
        }
        temp = (voltage/2000.0)*100.0;
        return temp;

}
```

9. Source code in C for Wavelet-based control of temperature. Program name: *WDMC.C*

```
/*This programme incorporates Wavelet-based Dynamic Matrix Control of a process parameter x
through the manipulated variable m using wavelet based blocking and condensing methodology*/

#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<math.h>
#include<dos.h>
#include<time.h>

#define BASE 0x218  /* set base address */

void init_matrix(double **,int n, int m);
void unit_matrix(double **, int n);
void weighting_matrix(double **, int n, double msp);
void dynamic_matrix(double **a, int np, int nc, double *beta);
void print_matrix(double **,int n, int m);
void ludcmp(double **a, double **lo, int n);
void luforwdsb(double **lo, double *b, int n);
void lubksb(double **a, double *b, int n);
double **matrix_trps(double **a, int nrows,int ncols);
double **matrix_add(double **a, int r1,int c1,double **b,int r2,int c2);
void matrix_mult(double **a,int r1,int c1,double **b,int r2,int c2, double **c);
double **ainv_b(double **a, double **matb, int n);
void steady_state(double v0);
void send_output(double v0);
double read_temp();
struct final_vector{int nr; int *fivec;};
typedef struct final_vector fv;
fv interval_vector(int np, double *x, double dy);
void cond_proj_mat(int np, double **cpmat,fv fv1);
void bloc_proj_mat(int np, double **bpmat,fv fv1);

FILE *fp, *fp1, *fp2, *fp3;
clock_t start,end;
time_t init,final;

main()
```

```
{
        double *y,**cpmat,*u,**bpmat,**cpmata,**bpmatt;
        double **deltam,**lddeltam,**deltam_old,**a,**lda,**ldat,**ldata;
        double **ldwtmat,**ldinputmat;
        double **perrmat,**ldperrmat,**ldsysmat,*xol,*beta;
        double temp,set_pt, sum,ts, msp, c_op, init_c_op, el_t=0.0,error;
        double cwitemp,hwitemp,cwfr,sum1=0.0,s_time,tot_time,ise=0.0;
        int i,j,k,np,nc,nt;
        fv fvc,fvb;         /* Final vector containing nr and fivec in condensing and blocking
respectively.*/
        double deltay,deltau;
        clrscr();
        nt=180; /* Truncation Horizon, which covers 90 to 95% of the steady state response */
        np=32;              /* Projection Horizon, which covers 90 to 95% of the steady state
response
                               Should be a power of 2 while applying wavelet*/
        nc=16;              /* Control Horizon, typically set at 50% of NP
                               Should be a power of 2 while applying wavelet*/
        deltay = 0.08;  /* Design parameter, dy for condensing and du for blocking*/
        deltau = 0.15;
        s_time=1.0;     /* Sampling time in seconds */
        tot_time=301.0; /* Time of operation */
        msp = 2000.0;  /*Move supression parameter, should be a positive real number */

        cwitemp=31.0;  /* Cold water inlet temp in degree C */
        hwitemp=80.0;  /* Hot water inlet temp in degree C, +_4 */
        cwfr=200.0;        /* Cold water flow rate in cc/min  */

        fp=fopen("c:\\tc3\\bin\\wdmcnt6.txt","a+");
        fprintf(fp,"HARE KRISHNA\n");
        fprintf(fp,"WAVELET BASED DMC,File: dmc1.c\n");
        fprintf(fp, "DATE: 15 OCT 2004\n");
        fprintf(fp,"Dynamic Matrix Control of the heat exchanger\n");
        fprintf(fp,"using wavelet based blocking and condensing\n");
        fprintf(fp,"Cold water inlet temperature =%lf\n",cwitemp);
        fprintf(fp,"Cold water flow rate (initial)=%lf\n",cwfr);
        fprintf(fp,"Cold water flow rate (final)= 200cc/min\n");
        fprintf(fp,"Hot water inlet temp =%lf\n",hwitemp);
        fprintf(fp,"Truncation Horizon, NT =%d\n",nt);
        fprintf(fp,"Projection Horizon, NP =%d\n",np);
        fprintf(fp,"Control Horizon, NC=%d\n",nc);
        fprintf(fp,"Condensing design parameter, dy=%lf\n",deltay);
        fprintf(fp,"Blocking design parameter, du=%lf\n",deltau);
        fprintf(fp,"Move supression parameter, msp=%lf\n",msp);
        fprintf(fp,"Sampling time (seconds)=%lf\n",s_time);
        fprintf(fp,"Total time (seconds)=%lf\n",tot_time);

        beta =(double *)malloc(nt*sizeof(double));
        xol =(double *)malloc(np*sizeof(double));

        y = (double *)malloc(sizeof(double)*np);
```

```
u = (double *)malloc(sizeof(double)*nc);

fp1 = fopen("c:\\tc3\\bin\\dmcstep1.txt","r");
fp2 = fopen("c:\\tc3\\bin\\condcoef.txt","r");
fp3 = fopen("c:\\tc3\\bin\\bloccoef.txt","r");
fflush(stdin);
for(i=0; i<nt; i++)
{
        fscanf(fp1,"%lf", &beta[i]);
}
fflush(stdin);
for(i=0; i<np; i++)
{
        fscanf(fp2,"%lf",&y[i]);
}
fflush(stdin);
for(i=0; i<nc; i++)
{
        fscanf(fp3,"%lf",&u[i]);
}
fvc=interval_vector(np,y,deltay);
fvb=interval_vector(nc,u,deltau);

cpmat = (double **)malloc(sizeof(double *)*fvc.nr);
bpmat = (double **)malloc(sizeof(double *)*fvb.nr);
a = (double **)malloc(sizeof(double *) * np);
cpmata = (double **)malloc(sizeof(double *) * fvc.nr);
bpmatt = (double **)malloc(sizeof(double *) * nc);
lda = (double **)malloc(sizeof(double *) * fvc.nr);
ldat = (double **)malloc(sizeof(double *) * fvb.nr);
ldata = (double **)malloc(sizeof(double *) * fvb.nr);
ldwtmat = (double **)malloc(sizeof(double *) * fvb.nr);
ldinputmat = (double **)malloc(sizeof(double *) * fvb.nr);
ldsysmat = (double **)malloc(sizeof(double *) * fvb.nr);
deltam = (double **)malloc(sizeof(double *) * nc);
lddeltam = (double **)malloc(sizeof(double *) * fvb.nr);
deltam_old = (double **)malloc(sizeof(double *) * nt);
perrmat = (double **)malloc(sizeof(double *) * np);
ldperrmat = (double **)malloc(sizeof(double *) * fvc.nr);

for(i=0; i<fvc.nr; i++)
{
        cpmat[i] = (double *)malloc(sizeof(double)*np);
        cpmata[i] = (double *)malloc(sizeof(double)*nc);
        lda[i] = (double *)malloc(sizeof(double)*fvb.nr);
        ldperrmat[i] = (double *)malloc(sizeof(double)*1);
}
for(i=0; i<fvb.nr; i++)
{
        bpmat[i] = (double *)malloc(sizeof(double)*nc);
        ldat[i] = (double *)malloc(sizeof(double)*fvc.nr);
```

```
        ldata[i] = (double *)malloc(sizeof(double)*fvb.nr);
        ldwtmat[i] = (double *)malloc(sizeof(double)*fvb.nr);
        ldinputmat[i] = (double *)malloc(sizeof(double)*fvb.nr);
        ldsysmat[i] = (double *)malloc(sizeof(double)*fvc.nr);
        lddeltam[i] = (double *)malloc(sizeof(double)*1);
}
for(i=0; i<np; i++)
{
        a[i] = (double *)malloc(sizeof(double)*nc);
        perrmat[i] = (double *)malloc(sizeof(double)*1);
}
for(i=0; i<nt; i++)
{
    deltam_old[i] = (double *)malloc(sizeof(double)*1);
}
for(i=0;i<nc;i++)
{
        bpmatt[i] = (double *)malloc(sizeof(double)*fvb.nr);
        deltam[i] = (double *)malloc(sizeof(double)*1);
}

printf("Enter the initial controller position: ");
scanf("%lf",&init_c_op);
printf("HARE KRISHNA !\n");
printf("<HIT ANY KEY TO START>\n");
fprintf(fp,"Initial controller position=%f\n",init_c_op);
steady_state(init_c_op);
//fprintf(fp,"STEADY STATE VALUES IN EVERY 1 SECOND\n");
while(!kbhit())
{
        printf("WAIT - SYSTEM BEING BROUGHT TO STEADY STATE\n\n\n");
        printf("Check connections before hitting any key\n");
        printf("Hit any key when steady state is reached\n\n\n");
        temp = read_temp();
        printf("Temperature=%f\n\n\n",temp);
        //fprintf(fp,"%lf\n",temp);
        delay(2000);
}
//fprintf(fp,"DYNAMIC VALUES\n");
printf("Enter the set point in cold water outlet temperature:");
scanf("%lf",&set_pt);
fprintf(fp,"Steady state cold water outlet temperature =%lf\n",temp);
fprintf(fp,"Set point=%lf\n",set_pt);
fprintf(fp,"\n\nTime(sec)\tC_ OP\t\tTemp(C)\t\tTime spent(sec)\n");
/*Steady state values*/
/*The current instant is 0th instant*/
init_matrix(deltam_old,nt,1);
for(k=0; k<nt; k++)
{
        deltam_old[k][0]=0.0;    /* Control matrix, deltaM_old */
}
```

```
init_matrix(perrmat,np,1);
for(i=0; i<np; i++)
{
        xol[i]=temp;            /* Open loop values of x, temp */
        for(j=0; j<1; j++)
        {
                perrmat[i][j]=set_pt-xol[i]; /*Predicted error Matrix*/
        }
}
cond_proj_mat(np,cpmat,fvc); /*Getting cpmat and bpmat*/
bloc_proj_mat(nc,bpmat,fvb);

/* Initialization of the matrices */
init_matrix(a,np,nc);
init_matrix(cpmata,fvc.nr,nc);
init_matrix(bpmatt,nc,fvb.nr);
init_matrix(lda,fvc.nr,fvb.nr);
init_matrix(ldat,fvb.nr,fvc.nr);
init_matrix(ldata,fvb.nr,fvb.nr);
init_matrix(ldwtmat,fvb.nr,fvb.nr);
init_matrix(ldinputmat,fvb.nr,fvb.nr);
init_matrix(ldsysmat,fvb.nr,fvc.nr);
//init_matrix(deltam,nc,1);
init_matrix(lddeltam,fvb.nr,1);

dynamic_matrix(a,np,nc,beta); /*Entering the values for dynamic matrix */
bpmatt = matrix_trps(bpmat,fvb.nr,nc);
matrix_mult(cpmat,fvc.nr,np,a,np,nc,cpmata);
matrix_mult(cpmata,fvc.nr,nc,bpmatt,nc,fvb.nr,lda);
ldat = matrix_trps(lda,fvc.nr,fvb.nr);
matrix_mult(ldat,fvb.nr,fvc.nr,lda,fvc.nr,fvb.nr,ldata);
weighting_matrix(ldwtmat,fvb.nr,msp); /* Weighting matrix with same msp value in all
                                         the diagonal element */
ldinputmat = matrix_add(ldata,fvb.nr,fvb.nr,ldwtmat,fvb.nr,fvb.nr);   /*Input matrix*/
ldsysmat = ainv_b(ldinputmat,ldat,fvb.nr);   /*Lower dimensional System Matrix*/

/*Calculating the future control moves, deltam[0] to deltam[NC-1]*/

while(el_t <= tot_time)
{
        init=time(NULL);
        start = clock();
        error = set_pt - temp;          /*Actual error*/
        ise = ise + pow(error,2)*s_time;     /*Integral of the square error*/
        matrix_mult(cpmat,fvc.nr,np,perrmat,np,1,ldperrmat);
        /* Control action */
        matrix_mult(ldsysmat,fvb.nr,fvc.nr,ldperrmat,fvc.nr,1,lddeltam); /*Control
                                                                  matrix, deltaM*/
        /*Implement the first control move from the optimum future control
        moves, that is deltam[0][0]*/
```

```
        sum1 += lddeltam[0][0];   /*Summation of all "change in control moves"*/
        c_op = init_c_op + sum1;  /*Control action to be taken*/
        if(c_op > 2.5)
        {
                c_op = 2.5;
        }
        else if(c_op < 0.0)
        {
                c_op = 0.0;
        }
        printf("C_OP=%lf\t",c_op);
        send_output(c_op);

        /*Shift the control moves and calculate the openloop and error
        values*/
        for(i=nt-1; i>=1; i--)
        {
                deltam_old[i][0]=deltam_old[i-1][0];
        }
        deltam_old[0][0]=lddeltam[0][0];     /* Shifting the control measures */
        temp = read_temp();                  /*On-line temp value*/
        //error = set_pt - temp;             /*Actual error*/

        for(i=0;i<np;i++)
        {
                sum=0.0;
                for(k=0; k<nt; k++)
                {
                 sum=sum+(beta[i+1+k]-beta[1+k])*deltam_old[k][0];
                }
                xol[i] = temp + sum;                  /*Open loop response*/
                perrmat[i][0]=set_pt-xol[i];     /*Projected error vector/matrix*/
        }
        end = clock();
        ts=(end-start)/CLK_TCK;
        if(ts<s_time)
        {
                delay((int)((s_time-ts)*1000));
        }
        final=time(NULL);
        el_t += difftime(final,init); /* Time lapsed as an integer count */
        printf("Error=%lf\tTime=%lf\tTemp=%lf\n",error,el_t,temp);
        fprintf(fp,"%lf\t %lf\t\t %lf\t\t %lf\n",el_t,c_op,temp,ts);

    }
    fprintf(fp,"\nISE=%lf\n",ise);
    fclose(fp);
    fclose(fp1);
    fclose(fp2);
    return 0;
}
```

```
void dynamic_matrix(double **a, int np, int nc, double *beta)
{
        int i,k;
        for(i=0; i<np; i++)
        {
                for(k=0; k<nc; k++)
                {
                        if(i<k)
                        {
                                a[i][k]=0.0;
                        }
                        else
                        {
                                a[i][k]=beta[i+1-k];
                        }
                }
        }
}

fv interval_vector(int np, double *x, double dy)
{
        fv fv1;
        int i,j,k,l,m,n,o,p,lmax,nmax;
        int **check, **ivec;
        double **s,**ela;
        double sum, sumsqr;
        lmax = log (np)/ log (2);
        nmax = np/2;
        p = np+nmax; /*Maximum dimension of the interval vector*/

        s = (double **)malloc(sizeof(double *)*(lmax+1));
        ela = (double **)malloc(sizeof(double *)*(lmax+1));
        check = (int **)malloc(sizeof(int *)*(lmax+1));
        ivec = (int **)malloc(sizeof(int *)*(lmax+1));

        for(l=0; l<=lmax; l++)
        {
                ivec[l] = (int *)malloc(sizeof(int)*p);
        }
        /*Wavelet Packet Decomposition*/
        /*Initializing interval vector at level 0*/
        for(k=0; k<p; k++)
        {
                if(k<=np)
                {
                        ivec[0][k] = k;
                }
                else
                {
                        ivec[0][k] = np;
                }
```

```
}
for(l=1; l<=lmax; l++)
{
        m = pow(2,l);
        n = np/m;
        /*Dynamic mamory allocation*/
        s[l] = (double *)malloc(sizeof(double)*n);
        ela[l] = (double *)malloc(sizeof(double)*n);
        check[l] = (int *)malloc(sizeof(int)*n);

        /*Finding the smooth components, energy losses and interval vector
        in each level*/

        j = 0; /*J is the index for the Interval vector*/
        for(i=0; i<n; i++)
        {
                sum =0.0;
                sumsqr = 0.0;
                for(o=0; o<m; o++)
                {
                        sum = sum + x[m*i+o];
                        sumsqr = sumsqr + pow(x[m*i+o],2);
                }
                s[l][i] = sum/sqrt(m); /* Smooth or approximation coefficient*/
                ela[l][i] = sumsqr - pow(s[l][i],2); /*Energy loss in approximation*/
                if(ela[l][i] <= dy * sumsqr/m) /* Energy criterion*/
                {
                        check[l][i] =1;
                        ivec[l][j] = m*i;
                        ivec[l][j+1] = m*(i+1);
                        j = j+2;
                }
                else
                {
                        check[l][i] = 0;
                        for(k=0; k<p; k++)
                        {
                                if(ivec[l-1][k] >= m*i)
                                {
                                        if(ivec[l-1][k] <= m*(i+1))
                                        {
                                                ivec[l][j] = ivec[l-1][k];
                                                j++;
                                        }
                                }
                        }
                }

        }
        /*Assigning remaining values as np*/
```

```
                for(k=0; k<=(p-j); k++)
                {
                        ivec[l][j+k] = np;
                }
                /*Condensing the repeated values */
                for(k=1; k<p; k++)
                {
                        if(ivec[l][k] == ivec[l][k-1])
                        {
                                for(o=0; o<(p-k); o++)
                                {
                                        ivec[l][k-1+o] = ivec[l][k+o];
                                }
                        }
                }

        }
        /* Finding the number of rows in the projection matrix */
        for(i=0; i<p; i++)
        {
                if(ivec[lmax][i] == np)
                {
                        fv1.nr = i;
                        break; /*Break when finding np for the first time*/
                }
        }
        /*Final Interval Vector*/
        fv1.fivec = (int *)malloc(sizeof(int)*(fv1.nr+1));
        for(i=0; i<=fv1.nr; i++)
        {
                fv1.fivec[i] = ivec[lmax][i];
        }
        return fv1;
}

void cond_proj_mat(int np, double **cpmat,fv fv1)
{
        int i,j,k,m;
        /*Initializing projection matrix*/
        init_matrix(cpmat,fv1.nr,np);
        /*Condensing Projection Matrix */
        for(i=0; i<fv1.nr; i++)
        {
                m = fv1.fivec[i+1] - fv1.fivec[i];
                for(j=0; j<m; j++)
                {
                        k = fv1.fivec[i] + j;
                        cpmat[i][k] = (double)1/m;
                }
        }
}
```

```
void bloc_proj_mat(int np, double **bpmat,fv fv1)
{
        int i,j,k,m;
        /*Initializing projection matrix*/
        init_matrix(bpmat,fv1.nr,np);
        /*Blocking projection matrix*/
        for(i=0; i<fv1.nr; i++)
        {
                m = fv1.fivec[i+1] - fv1.fivec[i];
                for(j=0; j<m; j++)
                {
                        k = fv1.fivec[i] + j;
                        bpmat[i][k] = (double)(m-j)/m;
                }
        }
}

void init_matrix(double **a,int n,int m)
{
        int i,j;

        for(i=0;i<n;i++)
        {
                for(j=0;j<m;j++)
                {
                        a[i][j] = 0.0;
                }
        }
}

void print_matrix(double **a,int n, int m)
{
        int i,j;
        for(i=0;i<n;i++)
        {
                for(j=0;j<m;j++)
                {
                        printf("\t%lf",a[i][j]);
                }
                printf("\n");
        }
}

void matrix_mult(double **a, int r1,int c1,double **b,int r2,int c2, double **c)
{
        int i,j,k;
        if(c1 != r2)
        {
                c = NULL;
        }
```

```
void bloc_proj_mat(int np, double **bpmat,fv fv1)
{
        int i,j,k,m;
        /*Initializing projection matrix*/
        init_matrix(bpmat,fv1.nr,np);
        /*Blocking projection matrix*/
        for(i=0; i<fv1.nr; i++)
        {
                m = fv1.fivec[i+1] - fv1.fivec[i];
                for(j=0; j<m; j++)
                {
                        k = fv1.fivec[i] + j;
                        bpmat[i][k] = (double)(m-j)/m;
                }
        }
}

void init_matrix(double **a,int n,int m)
{
        int i,j;

        for(i=0;i<n;i++)
        {
                for(j=0;j<m;j++)
                {
                        a[i][j] = 0.0;
                }
        }
}

void print_matrix(double **a,int n, int m)
{
        int i,j;
        for(i=0;i<n;i++)
        {
                for(j=0;j<m;j++)
                {
                        printf("\t%lf",a[i][j]);
                }
                printf("\n");
        }
}

void matrix_mult(double **a, int r1,int c1,double **b,int r2,int c2, double **c)
{
        int i,j,k;
        if(c1 != r2)
        {
                c = NULL;
        }
```

```
        for(i=0;i<r1;i++)
        {
                for(j=0;j<c2;j++)
                {
                        c[i][j] = 0;
                        for(k = 0;k<r2;k++)
                        {
                                c[i][j] += a[i][k] * b[k][j];
                        }
                }
        }
}

double **matrix_add(double **a, int r1,int c1,double **b,int r2,int c2)
{
        int i,j,k;
        double **c;
        if(r1 != r2 || c1 != c2)
        {
                return NULL;
        }
        c = (double **)malloc(sizeof(double *)*r1);
        for(i=0;i<r1;i++)
        {
                c[i] = (double *)malloc(sizeof(double)*c2);
        }
        for(i=0;i<r1;i++)
        {
                for(j=0;j<c2;j++)
                {
                        c[i][j] = a[i][j] + b[i][j];
                }
        }
        return c;
}

double **ainv_b(double **a, double **matb, int n)
{
        int i,j;
        double *b, **ainvb, **lo;
        b = (double *)malloc(sizeof(double) * n);
        ainvb = (double **)malloc(sizeof(double *) * n);
        lo = (double **)malloc(sizeof(double *) * n);
        for(i=0;i<n;i++)
        {
                ainvb[i] = (double *)malloc(sizeof(double)*n);
                lo[i] = (double *)malloc(sizeof(double)*n);
        }
        init_matrix(ainvb,n,n);
        unit_matrix(lo,n);
        ludcmp(a, lo, n);
```

```
        for(j=0; j<n; j++)
        {
                for(i=0; i<n; i++)
                {
                        b[i]=matb[i][j];
                }
                luforwdsb(lo, b, n);
                lubksb(a, b, n);
                for(i=0; i<n; i++)
                {
                        ainvb[i][j]=b[i];
                }
        }
        return ainvb;
}

void ludcmp(double **a, double **lo, int n)
/*This subroutine decomposes a given square matrix of dimension n to its
upper triangular matrix a and lower triangular matrix lo. This routine fails
when the pivot element happen to be zero or near zero. Initially lo is to be
defined as an identity matrix.*/
{
        int i,j,k;
        double mfactor;
        for(k=1;k<n;k++)
        {
                for(i=k;i<n;i++)
                {
                        mfactor=a[i][k-1]/a[k-1][k-1];
                        lo[i][k-1]=mfactor;
                        for(j=k-1;j<n;j++)
                        {
                            a[i][j]=a[i][j]-mfactor*a[k-1][j];
                        }
                }
        }
}

void luforwdsb(double **lo, double *b, int n)
{
        /*Forward sweep*/
        int i,j;
        double sum;
        for(i=1; i<n; i++)
        {
                sum=0.0;
                for(j=0;j<i;j++)
                {
                        sum=sum+lo[i][j]*b[j];
                }
                b[i]=b[i]-sum;
```

```
        }
}

void lubksb(double **a, double *b, int n)
{
        int i,j;
        double sum;
        /*Backward sweep*/
        /*The solution comes as a 'b' vector. If you want as x' vector, you
        can simply put x in the left hand side and define at the beginning.*/
        b[n-1]=b[n-1]/a[n-1][n-1]; /*The last element*/
        for(i=n-2;i>=0;i--)
        {
                sum=0.0:
                for(j=i+1;j<n;j++)
                {
                        sum=sum+a[i][j]*b[j];
                }
                b[i]=(b[i]-sum)/a[i][i];
        }
}

void unit_matrix(double **lo, int n)
{
        int i,j;
        for(i=0; i<n; i++)
        {
                for(j=0; j<n; j++)
                {
                        if(i==j)
                        {
                                lo[i][j]=1.0;
                        }
                        else
                        {
                                lo[i][j]=0.0;
                        }
                }
        }
}

void weighting_matrix(double **lo, int n, double msp)
{
        int i,j;
        for(i=0; i<n; i++)
        {
                for(j=0; j<n; j++)
                {
                        if(i==j)
                        {
                                lo[i][j] = msp;
```

```
                    }
                    else
                    {
                            lo[i][j]=0.0;
                    }
            }
        }
}

double **matrix_trps(double **a, int nrows,int ncols)
{
        int i,j;
        double **at;
        at=(double **)malloc(sizeof(double *)*ncols);
        for(i=0; i<ncols; i++)
        {
                at[i]=(double *)malloc(sizeof(double)*nrows);
        }
        for(i=0; i<ncols; i++)
        {
                for(j=0; j<nrows; j++)
                {
                        at[i][j]=a[j][i];
                }
        }
        return at;
}

void steady_state(double v0)
{
        int n,lb,hb;
        while(!kbhit())
        {
                n = (int) ((v0+5.0)*4095.0/10);
                lb = n & 0xff;
                hb = (int) (n-lb)/256.0;
                outportb (BASE + 2, lb);
                outportb (BASE + 3, hb);
        }
        getch();
        clrscr();
}

void send_output(double v0)
{
        int n,lb,hb;
        n = (int) ((v0 + 5.0)*4095.0/10.0);
        lb = n & 0xff;
        hb = (int) (n - lb)/256.0;
        outportb (BASE + 2, lb);
        outportb (BASE + 3, hb);
```

```
}

double read_temp()
{
        int i,lb,hb,status,adc_no;
        float temp,voltage=0.0;
        for(i=0; i<2000; i++)
        {
                outportb (BASE+0, 0);
                outportb (BASE+1, 0);
                status = inport (BASE+0);
                while(status < 128)
                        status = inport (BASE+0);
                lb = inportb (BASE+2);
                hb = inportb (BASE+3);
                hb = hb & 0x0f;
                adc_no = lb + 256*hb;
                voltage += (adc_no/4096.0)*10.0 - 5.0;
        }
        temp = (voltage/2000.0)*100.0;
        return temp;
}
```

APPENDIX II

EXPERIMENTAL DATA

List of Experimental Data **Page No.**

1. Process reaction curve of the heat exchanger process for a step change of 150 cc/min in the hot water flow rate keeping cold water flow rate constant at 200 cc/min (Servo Problem).

Time(s)	Temp (°C)
0	25.01538
1	25.07446
2	25.44959
3	25.9917
4	26.52673
5	27.18372
6	27.9306
7	28.77478
8	29.62244
9	30.25098
10	30.78845
11	31.2876
12	31.85291
13	32.33606
14	32.90186
15	33.55725
16	33.8877
17	34.15747
18	34.36621
19	34.6532
20	34.93799
21	35.18994
22	35.49805
23	35.77222
24	35.98023
25	36.22131
26	36.46436
27	36.77637
28	37.07642
29	37.34558
30	37.56152
31	37.70239
32	37.8208
33	37.99219
34	38.12073
35	38.36536
36	38.4552
37	38.59766
38	38.76355
39	38.93408
40	39.09497
41	39.2052
42	39.44019
43	39.5846
44	39.73108
45	39.85071
46	40.0072
47	40.10999
48	40.27539
49	40.46826
50	40.6272
51	40.82336
52	40.98914
53	41.18213
54	41.25427
55	41.38574
56	41.45117
57	41.46045
58	41.48364
59	41.5791
60	41.68005
61	41.76245
62	41.82947
63	41.87073
64	41.91992
65	42.00476
66	42.09583
67	42.18848
68	42.26868
69	42.29529
70	42.31165
71	42.34692
72	42.35559
73	42.43567
74	42.49536
75	42.56555
76	42.62329
77	42.71338
78	42.78674
79	42.90698
80	42.93445
81	42.99939
82	43.05664
83	43.0813
84	43.13037
85	43.19714
86	43.24451
87	43.28967
88	43.31189
89	43.32227
90	43.39905
91	43.40295
92	43.44287
93	43.50574
94	43.55627
95	43.59436
96	43.62708
97	43.64343
98	43.70142
99	43.71436
100	43.76172
101	43.79077
102	43.82678
103	43.86536
104	43.94739
105	43.93115
106	43.99719
107	44.00794
108	44.05042
109	44.06006
110	44.10217
111	44.11548
112	44.14624
113	44.16504
114	44.17932
115	44.20239
116	44.22058
117	44.27441
118	44.2721
119	44.28503
120	44.37378
121	44.34045
122	44.41809
123	44.41357
124	44.41797
125	44.48047
126	44.45044
127	44.50671
128	44.50537
129	44.53003
130	44.5708
131	44.61951
132	44.61169
133	44.64514
134	44.65601
135	44.67261
136	44.71314
137	44.74109
138	44.7594
139	44.75769
140	44.80689
141	44.77332
142	44.79321
143	44.77979
144	44.80103
145	44.80713
146	44.79504
147	44.79224
148	44.776
149	44.76184
150	44.78479
151	44.81201
152	44.80798
153	44.79321
154	44.81421
155	44.79932
156	44.81287
157	44.79907
158	44.83435
159	44.82459
160	44.85535
161	44.87048
162	44.89453
163	44.90039
164	44.92163
165	44.90125
166	44.90088

167	44.95935	187	45.11707	207	45.08728	227	45.06458
168	44.94055	188	45.12134	208	45.1001	228	45.03638
169	44.96863	189	45.11743	209	45.06616	229	45.05542
170	44.98047	190	45.11438	210	45.08313	230	45.06567
171	44.97437	191	45.19202	211	45.10571	231	45.0166
172	44.94702	192	45.15723	212	45.06982	232	45.01001
173	44.98572	193	45.14478	213	45.05933	233	45.04065
174	44.99487	194	45.14038	214	45.0658	234	45.09692
175	45.00037	195	45.14587	215	45.07678	235	45.02722
176	45.06238	196	45.13879	216	45.07666	236	45.00073
177	45.02295	197	45.1676	217	45.07983	237	45.02673
178	45.07556	198	45.1239	218	45.01465	238	45.01074
179	45.07983	199	45.12647	219	45.05469	239	44.99426
180	45.10327	200	45.1333	220	45.03894	240	45.00305
181	45.06165	201	45.11633	221	45.07861	241	44.97461
182	45.10559	202	45.13074	222	45.06592	242	44.8949
183	45.06897	203	45.11694	223	45.07922	243	44.90991
184	45.10034	204	45.10083	224	45.06226	244	44.83801
185	45.11401	205	45.11353	225	45.06628		
186	45.10169	206	45.08997	226	45.06116		

2. Process reaction curve of the heat exchanger process for a step change of –130 cc/min in the cold water flow rate, keeping hot water flow rate constant at 85 cc/min (regulatory problem).

Time (s)	Temp (°C)						
0	40.44434	20	45.11951	42	48.55042	64	50.51514
1	40.47351	21	45.36511	43	48.70288	65	50.63245
2	40.36731	22	45.54504	44	48.88806	66	50.70349
3	40.43115	23	45.7096	45	48.95227	67	50.82849
4	40.6001	24	45.89868	46	49.08252	68	50.94605
5	40.948	25	46.04993	47	49.18945	69	51.01318
6	41.31787	26	46.17969	48	49.3075	70	51.073
7	41.54456	27	46.41956	49	49.40955	71	51.17627
8	41.79456	28	46.61926	50	49.46558	72	51.23145
9	42.09924	29	46.76599	51	49.53906	73	51.33899
10	42.39258	30	46.95996	52	49.58923	74	51.50708
11	42.66589	31	47.09033	53	49.65796	75	51.62158
12	42.90637	32	47.23047	54	49.6908	76	51.70349
13	43.11926	33	47.34082	55	49.82068	77	51.82971
14	43.33447	34	47.51697	56	49.896	78	51.88538
15	43.5564	35	47.56506	57	49.97669	79	51.97022
16	43.84363	36	47.70252	58	50.04346	80	52.08716
17	44.16504	37	47.85803	59	50.17859	81	52.13318
18	44.46875	38	47.98377	60	50.25708	82	52.24231
19	44.76831	39	48.13525	61	50.35962	83	52.30689
		40	48.24634	62	50.36353	84	52.29712
		41	48.41602	63	50.43884	85	52.36853

86	52.55579
87	52.68518
88	52.74109
89	52.88171
90	52.98486
91	53.05762
92	53.12329
93	53.12378
94	53.25794
95	53.36182
96	53.3147
97	53.3042
98	53.38025
99	53.49426
100	53.55139
101	53.60071
102	53.58887
103	53.5282
104	53.60156
105	53.62354
106	53.69128
107	53.724
108	53.74841
109	53.73901
110	53.74707
111	53.81604
112	53.80042
113	53.84802
114	53.96338
115	54.05823
116	54.08215
117	54.19739
118	54.25855
119	54.30786
120	54.31873
121	54.40149
122	54.40234
123	54.45972
124	54.45288
125	54.55847
126	54.58801
127	54.61548
128	54.70813
129	54.78503
130	54.81018
131	54.84595
132	54.89099
133	54.96606
134	54.98853
135	55.01599
136	55.01257
137	55.06421
138	55.16284
139	55.2207
140	55.2052
141	55.26746
142	55.2793
143	55.29773
144	55.30591
145	55.25232
146	55.37354
147	55.34387
148	55.36694
149	55.37976
150	55.40894
151	55.3949
152	55.44128
153	55.51331
154	55.56677
155	55.58301
156	55.62036
157	55.66394
158	55.73621
159	55.74573
160	55.77002
161	55.83106
162	55.82849
163	55.9104
164	55.92749
165	55.90906
166	55.97583
167	56.02832
168	56.05139
169	56.10132
170	56.18897
171	56.24548
172	56.20483
173	56.26209
174	56.32336
175	56.32886
176	56.40857
177	56.41748
178	56.4624
179	56.49023
180	56.52283
181	56.57544
182	56.59436
183	56.56262
184	56.54834
185	56.50427
186	56.4906
187	56.50549
188	56.49927
189	56.47437
190	56.53772
191	56.51331
192	56.54492
193	56.59534
194	56.64722
195	56.64539
196	56.6709
197	56.74426
198	56.72022
199	56.76367
200	56.78394
201	56.77698
202	56.84668
203	56.87695
204	56.87085
205	56.83154
206	56.9259
207	56.91931
208	56.94605
209	56.95105
210	56.90857
211	56.97498
212	57.04077
213	57.0647
214	57.06543
215	57.08545
216	57.10767
217	57.10693
218	57.11499
219	57.14075
220	57.12427
221	57.12524
222	57.11987
223	57.15222
224	57.12183
225	57.09241
226	57.08984
227	57.06458
228	57.07727
229	57.08276
230	57.0658
231	57.06567
232	57.04932
233	57.06262
234	57.02417
235	57.03381
236	57.05017
237	57.05847
238	57.08398
239	57.02637
240	57.02649
241	57.00403
242	56.9386

3. Closed loop response of the heat exchanger process using PID controller for a step change of 10°C in the set point (35°C to 45°C).

Time (s)	HWFR (cc/min)	Temp (°C)
0	10.2345	34.98123
1	26.66667	35.01452
2	42.63541	35.06979
3	57.84097	35.20427
4	72.22768	35.39822
5	85.7493	35.64636
6	98.36869	35.94336
7	110.0575	36.28392
8	120.7956	36.66279
9	130.571	37.07478
10	139.3791	37.51484
11	147.2225	37.97802
12	154.1103	38.45958

13	160.0575	38.95493	63	61.89175	45.33542	113	88.18224	45.10861
14	165.0848	39.45971	64	62.38958	45.22973	114	87.88628	45.1201
15	169.2181	39.96976	65	63.00967	45.12943	115	87.58715	45.13
16	172.4877	40.48118	66	63.73961	45.03487	116	87.28715	45.13833
17	174.9277	40.99031	67	64.56705	44.94631	117	86.98843	45.14515
18	176.5762	41.49374	68	65.47976	44.86399	118	86.69298	45.1505
19	177.4738	41.98835	69	66.46573	44.78806	119	86.40266	45.15445
20	177.6642	42.47126	70	67.51323	44.71864	120	86.11914	45.15706
21	177.1926	42.93988	71	68.61088	44.65576	121	85.84396	45.1584
22	176.1061	43.39188	72	69.7477	44.59945	122	85.5785	45.16855
23	174.4531	43.8252	73	70.91318	44.54966	123	85.32395	45.17869
24	172.2825	44.23806	74	72.09727	44.50631	124	85.08138	45.18884
25	169.6437	44.62893	75	73.29048	44.46927	125	84.85167	45.19898
26	166.5861	44.99653	76	74.48389	44.43839	126	84.63559	45.20913
27	163.1588	45.33984	77	75.66914	44.41346	127	84.43375	45.21927
28	159.41	45.65808	78	76.83852	44.39428	128	84.2466	45.22942
29	155.3873	45.95067	79	77.9849	44.3806	129	83.7974	45.23956
30	151.1369	46.21727	80	79.10181	44.37214	130	83.78816	45.24971
31	146.7034	46.45774	81	80.1834	44.36862	131	83.77891	45.25985
32	142.1299	46.67213	82	81.22446	44.36975	132	83.76967	45.27
33	137.4576	46.86067	83	82.22041	44.3752	133	83.76042	45.28014
34	132.7257	47.02375	84	83.16727	44.38464	134	83.75118	45.29029
35	127.9711	47.16192	85	84.06168	44.39776	135	83.74193	45.30043
36	123.2284	47.27587	86	84.90089	44.41422	136	83.73269	45.31058
37	118.5301	47.36639	87	85.68266	44.43367	137	83.74465	45.32072
38	113.9059	47.43441	88	86.40536	44.45579	138	83.75661	45.33087
39	109.3832	47.48094	89	87.06784	44.48024	139	84.07	45.34101
40	104.9869	47.50708	90	87.66946	44.50671	140	84.08015	45.35116
41	100.7394	47.51399	91	88.21006	44.53486	141	84.09029	45.3613
42	96.66029	47.5029	92	88.68989	44.5644	142	84.10044	45.37145
43	92.76705	47.47508	93	89.10962	44.59503	143	84.11058	45.38159
44	89.07452	47.43182	94	89.47031	44.62646	144	84.12073	45.39174
45	85.59518	47.37445	95	89.77333	44.65841	145	84.13087	45.40188
46	82.33924	47.3043	96	90.02039	44.69064	146	84.14102	45.41203
47	79.31463	47.22271	97	90.21347	44.72289	147	84.15116	45.42217
48	76.52716	47.131	98	90.35479	44.75494	148	84.16131	45.43232
49	73.9806	47.0305	99	90.44678	44.78659	149	84.17145	45.44246
50	71.67678	46.92246	100	90.49207	44.81763	150	84.1816	45.45261
51	69.61574	46.80816	101	90.49342	44.84789	151	84.19174	45.46275
52	67.79578	46.68879	102	90.45375	44.87721	152	84.20189	45.4729
53	66.21368	46.56554	103	90.37603	44.90545	153	84.21203	45.48304
54	64.86473	46.43951	104	90.26333	44.93247	154	84.22218	45.49319
55	63.74294	46.31177	105	90.11874	44.95819	155	84.23232	45.50333
56	62.8411	46.18333	106	89.9454	44.98249	156	84.24247	45.5034
57	62.15096	46.05512	107	89.74639	45.00531	157	84.25261	45.49362
58	61.66333	45.92803	108	89.52482	45.02658	158	84.26276	45.48384
59	61.36823	45.80287	109	89.28372	45.04626	159	84.2729	45.47406
60	61.25497	45.68038	110	89.02607	45.06432	160	84.28305	45.46428
61	61.31231	45.56125	111	88.75476	45.08073	161	84.29319	45.45451
62	61.52858	45.44608	112	88.47259	45.09549	162	84.30334	45.44473

163	84.31348	45.43495	209	83.91334	44.98511	255	83.82817	45.10253
164	84.32363	45.42517	210	83.90356	44.97533	256	83.83865	45.11301
165	84.33377	45.41539	211	83.89379	44.96556	257	83.84913	45.12349
166	84.33384	45.40561	212	83.88401	44.95578	258	83.85961	45.13397
167	84.32406	45.39583	213	83.87423	44.946	259	83.87009	45.14445
168	84.31428	45.38605	214	83.86445	44.93622	260	83.88057	45.15493
169	84.3045	45.37627	215	83.85467	44.92644	261	83.89105	45.16541
170	84.29472	45.36649	216	83.84489	44.91666	262	83.90153	45.17589
171	84.28495	45.35672	217	83.83511	44.90688	263	83.91201	45.18637
172	84.27517	45.34694	218	83.82533	44.8971	264	83.92249	45.19685
173	84.26539	45.33716	219	83.81555	44.88732	265	83.93297	45.20733
174	84.25561	45.32738	220	83.80577	44.87754	266	83.94345	45.19809
175	84.24583	45.3176	221	83.796	44.86777	267	83.95393	45.18884
176	84.23605	45.30782	222	83.78622	44.85799	268	83.96441	45.1796
177	84.22627	45.29804	223	83.77644	44.84821	269	83.97489	45.17035
178	84.21649	45.28826	224	83.76666	44.83843	270	83.98537	45.16111
179	84.20671	45.27848	225	83.75688	44.82865	271	83.99585	45.15186
180	84.19693	45.2687	226	83.7471	44.81887	272	84.00633	45.14262
181	84.18716	45.25893	227	83.73732	44.80909	273	84.01681	45.13337
182	84.17738	45.24915	228	83.72754	44.81957	274	84.02729	45.12413
183	84.1676	45.23937	229	83.71776	44.83005	275	84.03777	45.11488
184	84.15782	45.22959	230	83.70798	44.84053	276	84.02853	45.10564
185	84.14804	45.21981	231	83.69821	44.85101	277	84.01928	45.09639
186	84.13826	45.21003	232	83.68843	44.86149	278	84.01004	45.08715
187	84.12848	45.20025	233	83.67865	44.87197	279	84.00079	45.0779
188	84.1187	45.19047	234	83.66887	44.88245	280	83.99155	45.06866
189	84.10892	45.18069	235	83.65909	44.89293	281	83.9823	45.05941
190	84.09914	45.17091	236	83.64931	44.90341	282	83.97306	45.05017
191	84.08937	45.16114	237	83.63953	44.91389	283	83.96381	45.04092
192	84.07959	45.15136	238	83.65001	44.92437	284	83.95457	45.03168
193	84.06981	45.14158	239	83.66049	44.93485	285	83.94532	45.02243
194	84.06003	45.1318	240	83.67097	44.94533	286	83.93608	45.01319
195	84.05025	45.12202	241	83.68145	44.95581	287	83.92683	45.00394
196	84.04047	45.11224	242	83.69193	44.96629	288	83.91759	44.9947
197	84.03069	45.10246	243	83.70241	44.97677	289	83.90834	44.98545
198	84.02091	45.09268	244	83.71289	44.98725	290	83.8991	44.97621
199	84.01113	45.0829	245	83.72337	44.99773	291	83.88985	44.96696
200	84.00135	45.07312	246	83.73385	45.00821	292	83.88061	44.95772
201	83.99158	45.06335	247	83.74433	45.01869	293	83.87136	44.94847
202	83.9818	45.05357	248	83.75481	45.02917	294	83.86212	44.93923
203	83.97202	45.04379	249	83.76529	45.03965	295	83.85287	44.92998
204	83.96224	45.03401	250	83.77577	45.05013	296	83.84363	44.92074
205	83.95246	45.02423	251	83.78625	45.06061	297	83.83438	44.91149
206	83.94268	45.01445	252	83.79673	45.07109	298	83.82514	44.90225
207	83.9329	45.00467	253	83.80721	45.08157	299	83.81589	44.91421
208	83.92312	44.99489	254	83.81769	45.09205	300	83.80665	44.92617

4. Closed-loop response of the heat exchanger process for a regulatory PID control, when a step disturbance of 100 cc/min is given in the cold water flow rate (regulatory problem).

Time (s)	HWFR (cc/min)	Temp (°C)
0	80.71193	44.9766
1	80.64336	44.98596
2	80.57478	44.99532
3	80.50621	45.00468
4	80.43764	45.01403
5	80.36906	45.02339
6	80.30049	45.03275
7	80.23192	45.04211
8	80.16334	45.05146
9	80.09477	45.06082
10	80.0262	45.07018
11	79.95762	45.07953
12	79.88905	45.08889
13	79.82048	45.09825
14	79.7519	45.10761
15	79.68333	45.11696
16	79.61476	45.12632
17	79.94203	45.12249
18	80.2693	45.11865
19	80.59657	45.11481
20	80.92384	45.11097
21	81.25111	45.10714
22	81.57838	45.1033
23	81.90565	45.09946
24	82.23292	45.09563
25	82.56019	45.09179
26	82.88747	45.08795
27	83.21474	45.08412
28	83.54201	45.08028
29	83.86928	45.07644
30	84.19655	45.0726
31	84.52382	45.06877
32	84.85109	45.06493
33	84.72541	45.06066
34	84.59972	45.0564
35	84.47403	45.05213
36	84.34835	45.04787
37	84.22266	45.0436
38	84.09697	45.03933
39	83.97129	45.03507
40	83.8456	45.0308
41	83.71991	45.02654
42	83.59423	45.02227
43	83.46854	45.018
44	83.34285	45.01374
45	83.21717	45.00947
46	83.09148	45.00521
47	82.96579	45.00094
48	82.84011	44.99668
49	83.00119	45.00523
50	83.16227	45.01379
51	83.32335	45.02235
52	83.48443	45.0309
53	83.64551	45.03946
54	83.80659	45.04802
55	83.96767	45.05657
56	84.12875	45.06513
57	84.28983	45.07369
58	84.45091	45.08224
59	84.61199	45.0908
60	84.77307	45.09936
61	84.93415	45.10791
62	85.09523	45.11647
63	85.25631	45.12503
64	85.41739	45.13358
65	85.54012	45.13414
66	85.66284	45.1347
67	85.78557	45.13527
68	85.9083	45.13583
69	86.03102	45.13639
70	86.15375	45.13695
71	86.27648	45.13751
72	86.39921	45.13807
73	86.52193	45.13863
74	86.64466	45.13919
75	86.76739	45.13975
76	86.89011	45.14031
77	87.01284	45.14088
78	87.13557	45.14144
79	87.2583	45.142
80	87.38102	45.14256
81	87.33137	45.14516
82	87.28171	45.14777
83	87.23205	45.15038
84	87.18239	45.15298
85	87.13273	45.15559
86	87.08308	45.1582
87	87.03342	45.1608
88	86.98376	45.16341
89	86.9341	45.16602
90	86.88444	45.16862
91	86.83479	45.17123
92	86.78513	45.17383
93	86.73547	45.17644
94	86.68581	45.17905
95	86.63615	45.18165
96	86.5865	45.18426
97	87.36561	44.98853
98	88.14472	44.79279
99	88.92384	44.59706
100	89.70295	44.40133
101	90.48207	44.2056
102	91.26118	44.00986
103	92.04029	43.81413
104	92.81941	43.6184
105	93.59852	43.42267
106	94.37764	43.22693
107	95.15675	43.0312
108	95.93587	42.83547
109	96.71498	42.63974
110	97.49409	42.444
111	98.27321	42.24827
112	99.05232	42.05254
113	100.2174	41.86503
114	101.3825	41.67753
115	102.5475	41.49002
116	103.7126	41.30252
117	104.8777	41.11501
118	106.0427	40.9275
119	107.2078	40.74
120	108.3729	40.55249
121	109.5379	40.36499
122	110.703	40.17748
123	111.8681	39.98998
124	113.0331	39.80247
125	114.1982	39.61497
126	115.3633	39.42746
127	116.5283	39.23995
128	117.6934	39.05245
129	118.6097	39.01854
130	119.5259	38.98464
131	120.4422	38.95073
132	121.3584	38.91683
133	122.2747	38.88292
134	123.191	38.84902
135	124.1072	38.81511
136	125.0235	38.78121
137	125.9398	38.7473
138	126.856	38.71339
139	127.7723	38.67949
140	128.6885	38.64558
141	129.6048	38.61168
142	130.5211	38.57777
143	131.4373	38.54387
144	132.3536	38.50996

145	134.0816	38.58378	200	240.9584	42.1155	255	256.1925	46.19456
146	135.8097	38.65759	201	243.0356	42.20501	256	254.7104	46.23167
147	137.5377	38.73141	202	245.1129	42.29452	257	253.1911	46.21148
148	139.2657	38.80522	203	247.1902	42.38403	258	251.6718	46.1913
149	140.9938	38.87904	204	249.2675	42.47354	259	250.1525	46.17111
150	142.7218	38.95285	205	251.3447	42.56305	260	248.6332	46.15093
151	144.4498	39.02667	206	253.422	42.65256	261	247.1139	46.13074
152	146.1779	39.10048	207	255.4993	42.74207	262	245.5946	46.11056
153	147.9059	39.17429	208	257.5766	42.83158	263	244.0754	46.09037
154	149.634	39.24811	209	258.796	42.94016	264	242.5561	46.07019
155	151.362	39.32192	210	260.0154	43.04875	265	241.0368	46.05
156	153.09	39.39574	211	261.2348	43.15734	266	239.5175	46.02982
157	154.8181	39.46955	212	262.4542	43.26593	267	237.9982	46.00963
158	156.5461	39.54337	213	263.6736	43.37452	268	236.4789	45.98945
159	158.2741	39.61718	214	264.8931	43.48311	269	234.9596	45.96926
160	160.0022	39.691	215	266.1125	43.5917	270	233.4403	45.94908
161	162.0357	39.74441	216	267.3319	43.70029	271	231.921	45.92889
162	164.0692	39.79781	217	268.5513	43.80888	272	230.4017	45.90871
163	166.1027	39.85122	218	269.7707	43.91747	273	230.2984	45.88436
164	168.1362	39.90463	219	270.9901	44.02606	274	230.1951	45.86
165	170.1697	39.95804	220	272.2096	44.13465	275	230.0918	45.83565
166	172.2032	40.01144	221	273.429	44.24324	276	229.9885	45.8113
167	174.2367	40.06485	222	274.6484	44.35183	277	229.8852	45.78695
168	176.2702	40.11826	223	275.8678	44.46042	278	229.7819	45.7626
169	178.3038	40.17167	224	277.0872	44.56901	279	229.6785	45.73825
170	180.3373	40.22507	225	277.1708	44.63582	280	229.5752	45.7139
171	182.3708	40.27848	226	277.2544	44.70263	281	229.4719	45.68955
172	184.4043	40.33189	227	277.338	44.76943	282	229.3686	45.66519
173	186.4378	40.3853	228	277.4216	44.83624	283	229.2653	45.64084
174	188.4713	40.4387	229	277.5052	44.90304	284	229.162	45.61649
175	190.5048	40.49211	230	277.5888	44.96985	285	229.0587	45.59214
176	192.5383	40.54552	231	277.6724	45.03666	286	228.9554	45.56779
177	194.5259	40.59889	232	277.756	45.10346	287	228.8521	45.54344
178	196.5135	40.65226	233	277.8396	45.17027	288	228.7487	45.51909
179	198.5012	40.70563	234	277.9232	45.23708	289	228.873	45.51126
180	200.4888	40.759	235	278.0068	45.30388	290	228.9973	45.50343
181	202.4764	40.81237	236	278.0904	45.37069	291	229.1216	45.49561
182	204.464	40.86574	237	278.1739	45.4375	292	229.2459	45.48778
183	206.4516	40.91911	238	278.2575	45.5043	293	229.3702	45.47995
184	208.4392	40.97248	239	278.3411	45.57111	294	229.4945	45.47212
185	210.4269	41.02585	240	278.4247	45.63791	295	229.6187	45.4643
186	212.4145	41.07922	241	276.9426	45.67502	296	229.743	45.45647
187	214.4021	41.13258	242	275.4604	45.71213	297	229.8673	45.44864
188	216.3897	41.18595	243	273.9783	45.74924	298	229.9916	45.44082
189	218.3773	41.23932	244	272.4961	45.78635	299	230.1159	45.43299
190	220.3649	41.29269	245	271.014	45.82346	300	230.2402	45.42516
191	222.3526	41.34606	246	269.5319	45.86057	301	230.3645	45.41734
192	224.3402	41.39943	247	268.0497	45.89768	302	230.4888	45.40951
193	226.4175	41.48894	248	266.5676	45.93479	303	230.613	45.40168
194	228.4947	41.57845	249	265.0854	45.9719	304	230.7373	45.39386
195	230.572	41.66796	250	263.6033	46.00901	305	230.8475	45.37003
196	232.6493	41.75747	251	262.1211	46.04612	306	230.9576	45.3462
197	234.7265	41.84698	252	260.639	46.08323	307	231.0677	45.32236
198	236.8038	41.93649	253	259.1568	46.12034	308	231.1778	45.29853
199	238.8811	42.026	254	257.6747	46.15745	309	231.2879	45.2747

310	231.3981	45.25087	365	238.8905	45.1443	420	236.7572	44.9892
311	231.5082	45.22704	366	239.0522	45.15327	421	236.8708	44.98956
312	231.6183	45.20321	367	239.2139	45.16224	422	236.9844	44.98992
313	231.7284	45.17938	368	239.3756	45.17121	423	237.098	44.99028
314	231.8386	45.15555	369	239.5022	45.15978	424	237.2116	44.99065
315	231.9487	45.13172	370	239.6288	45.14834	425	237.3252	44.99101
316	232.0588	45.10789	371	239.7554	45.13691	426	237.4388	44.99137
317	232.1689	45.08406	372	239.882	45.12548	427	237.5524	44.99173
318	232.2791	45.06023	373	240.0085	45.11404	428	237.666	44.99209
319	232.3892	45.0364	374	240.1351	45.10261	429	237.7796	44.99245
320	232.4993	45.01257	375	240.2617	45.09117	430	237.8932	44.99281
321	232.6585	45.01407	376	240.3883	45.07974	431	238.0068	44.99317
322	232.8176	45.01557	377	240.5149	45.0683	432	238.1204	44.99354
323	232.9767	45.01707	378	240.6415	45.05687	433	238.375	45.00325
324	233.1359	45.01858	379	240.768	45.04543	434	238.6297	45.01296
325	233.295	45.02008	380	240.8946	45.034	435	238.8844	45.02268
326	233.4542	45.02158	381	241.0212	45.02256	436	239.139	45.03239
327	233.6133	45.02308	382	241.1478	45.01113	437	239.3937	45.04211
328	233.7725	45.02458	383	241.2744	44.99969	438	239.6483	45.05182
329	233.9316	45.02608	384	241.401	44.98826	439	239.903	45.06154
330	234.0908	45.02758	385	240.9123	44.9854	440	240.1577	45.07125
331	234.2499	45.02909	386	240.4236	44.98254	441	240.4123	45.08096
332	234.4091	45.03059	387	239.935	44.97968	442	240.667	45.09068
333	234.5682	45.03209	388	239.4463	44.97682	443	240.9216	45.10039
334	234.7274	45.03359	389	238.9577	44.97396	444	241.1763	45.11011
335	234.8865	45.03509	390	238.469	44.9711	445	241.431	45.11982
336	235.0457	45.03659	391	237.9803	44.96825	446	241.6856	45.12954
337	235.1546	45.03603	392	237.4917	44.96539	447	241.9403	45.13925
338	235.2635	45.03547	393	237.003	44.96253	448	242.195	45.14896
339	235.3725	45.03492	394	236.5144	44.95967	449	241.9776	45.13954
340	235.4814	45.03436	395	236.0257	44.95681	450	241.7602	45.13012
341	235.5903	45.0338	396	235.537	44.95395	451	241.5429	45.1207
342	235.6992	45.03324	397	235.0484	44.95109	452	241.3255	45.11127
343	235.8082	45.03268	398	234.5597	44.94823	453	241.1082	45.10185
344	235.9171	45.03212	399	234.0711	44.94537	454	240.8908	45.09243
345	236.026	45.03156	400	233.5824	44.94252	455	240.6735	45.083
346	236.135	45.031	401	233.7524	44.94534	456	240.4561	45.07358
347	236.2439	45.03045	402	233.9224	44.94817	457	240.2387	45.06416
348	236.3528	45.02989	403	234.0925	44.951	458	240.0214	45.05474
349	236.4617	45.02933	404	234.2625	44.95383	459	239.804	45.04531
350	236.5707	45.02877	405	234.4325	44.95665	460	239.5867	45.03589
351	236.6796	45.02821	406	234.6026	44.95948	461	239.3693	45.02647
352	236.7885	45.02765	407	234.7726	44.96231	462	239.152	45.01704
353	236.9502	45.03662	408	234.9426	44.96514	463	238.9346	45.00762
354	237.1119	45.0456	409	235.1126	44.96796	464	238.7172	44.9982
355	237.2736	45.05457	410	235.2827	44.97079	465	238.5881	45.00186
356	237.4353	45.06354	411	235.4527	44.97362	466	238.459	45.00552
357	237.597	45.07251	412	235.6227	44.97645	467	238.3299	45.00918
358	237.7587	45.08149	413	235.7927	44.97927	468	238.2007	45.01284
359	237.9204	45.09046	414	235.9628	44.9821	469	238.0716	45.01649
360	238.0821	45.09943	415	236.1328	44.98493	470	237.9425	45.02015
361	238.2438	45.10841	416	236.3028	44.98776	471	237.8134	45.02381
362	238.4055	45.11738	417	236.4164	44.98812	472	237.6842	45.02747
363	238.5672	45.12635	418	236.53	44.98848	473	237.5551	45.03113
364	238.7289	45.13532	419	236.6436	44.98884	474	237.426	45.03479

475	237.2969	45.03845	517	240.9495	44.95299	559	242.5763	45.08952
476	237.1677	45.04211	518	241.2629	44.95113	560	242.7467	45.09421
477	237.0386	45.04577	519	241.5763	44.94926	561	242.6658	45.08897
478	236.9095	45.04943	520	241.8898	44.94739	562	242.5848	45.08373
479	236.7804	45.05309	521	242.2032	44.94552	563	242.5039	45.0785
480	236.6512	45.05675	522	242.5166	44.94365	564	242.4229	45.07326
481	236.5866	45.05333	523	242.83	44.94179	565	242.342	45.06802
482	236.5219	45.04991	524	243.1435	44.93992	566	242.261	45.06278
483	236.4573	45.04649	525	243.4569	44.93805	567	242.1801	45.05755
484	236.3926	45.04307	526	243.7703	44.93618	568	242.0991	45.05231
485	236.3279	45.03965	527	244.0838	44.93432	569	242.0182	45.04707
486	236.2633	45.03623	528	244.3972	44.93245	570	241.9372	45.04183
487	236.1986	45.03281	529	244.1236	44.93787	571	241.8563	45.03659
488	236.134	45.02939	530	243.8501	44.94328	572	241.7753	45.03136
489	236.0693	45.02597	531	243.5765	44.9487	573	241.6944	45.02612
490	236.0047	45.02255	532	243.303	44.95412	574	241.6135	45.02088
491	235.94	45.01913	533	243.0294	44.95954	575	241.5325	45.01564
492	235.8753	45.01571	534	242.7559	44.96495	576	241.4516	45.01041
493	235.8107	45.01229	535	242.4823	44.97037	577	241.3949	45.01149
494	235.746	45.00887	536	242.2088	44.97579	578	241.3382	45.01258
495	235.6814	45.00546	537	241.9352	44.98121	579	241.2816	45.01366
496	235.6167	45.00204	538	241.6617	44.98662	580	241.2249	45.01475
497	235.8521	44.99955	539	241.3881	44.99204	581	241.1683	45.01583
498	236.0874	44.99707	540	241.1146	44.99746	582	241.1116	45.01692
499	236.3228	44.99459	541	240.841	45.00288	583	241.055	45.018
500	236.5581	44.99211	542	240.5675	45.00829	584	240.9983	45.01909
501	236.7935	44.98963	543	240.2939	45.01371	585	240.9416	45.02017
502	237.0288	44.98715	544	240.0204	45.01913	586	240.885	45.02126
503	237.2642	44.98467	545	240.1908	45.02382	587	240.8283	45.02234
504	237.4995	44.98218	546	240.3612	45.02851	588	240.7717	45.02342
505	237.7349	44.9797	547	240.5316	45.03321	589	240.715	45.02451
506	237.9702	44.97722	548	240.702	45.0379	590	240.6584	45.02559
507	238.2056	44.97474	549	240.8724	45.04259	591	240.6017	45.02668
508	238.4409	44.97226	550	241.0428	45.04728	592	240.545	45.02776
509	238.6763	44.96978	551	241.2132	45.05198	593	240.5255	45.02794
510	238.9116	44.9673	552	241.3836	45.05667	594	240.506	45.02811
511	239.147	44.96481	553	241.554	45.06136	595	240.4865	45.02828
512	239.3823	44.96233	554	241.7244	45.06605	596	240.4669	45.02845
513	239.6958	44.96047	555	241.8948	45.07075	597	240.4474	45.02862
514	240.0092	44.9586	556	242.0651	45.07544	598	240.4279	45.02879
515	240.3226	44.95673	557	242.2355	45.08013	599	240.4083	45.02896
516	240.636	44.95486	558	242.4059	45.08482	600	240.3888	45.02914

5. Closed-loop response of the heat exchanger process using DMC for a step change of 10°C in the set point (35°C to 45°C).

Time (s)	HWFR (cc/min)	Temp (C)						
0	229.4895	35.05811	5	298.8846	37.21082	11	298.4132	42.33911
1	296.7175	35.32817	6	299.0789	37.96191	12	298.058	43.06384
2	297.8902	35.57026	7	299.8549	38.80554	13	297.7027	43.80774
3	298.0897	36.0636	8	299.9082	39.75061	14	297.3475	44.40405
4	298.6523	36.44592	9	299.1238	40.78503	15	296.9922	44.9895
			10	298.7685	41.68323	16	296.6369	45.47717

17	296.2817	45.78308	70	115.1617	45.76489	123	128.1486	44.98926
18	295.9264	46.02844	71	107.5116	45.86218	124	127.5962	45.02661
19	290.5712	46.30847	72	100.3016	45.96106	125	126.7764	45.03638
20	283.6418	46.55127	73	93.68787	46.04041	126	125.7593	45.06567
21	269.3758	46.75598	74	88.15148	46.14429	127	124.3217	45.09485
22	252.9998	46.88611	75	83.41624	46.17029	128	122.4989	45.09375
23	236.4904	47.02661	76	79.87152	46.21314	129	120.7101	45.10083
24	219.7378	47.13452	77	76.99343	46.16711	130	119.2431	45.11804
25	203.1284	47.31067	78	75.32774	46.1593	131	117.73	45.10779
26	185.7339	47.49402	79	74.22313	46.15906	132	116.7761	45.1178
27	167.3267	47.61353	80	73.45539	46.11597	133	115.7539	45.06384
28	149.2432	47.73792	81	73.20554	46.08411	134	115.7417	45.10999
29	131.5336	47.81714	82	73.01343	46.01233	135	115.2047	45.09192
30	115.2855	47.89087	83	73.11459	45.92883	136	115.1408	45.0979
31	100.2847	47.8927	84	73.36997	45.88025	137	115.1798	45.1023
32	87.92193	47.8385	85	73.64372	45.81641	138	115.6154	45.10217
33	78.11851	47.79834	86	74.38238	45.75293	139	115.9327	45.07117
34	70.25367	47.68481	87	75.45756	45.68494	140	116.527	45.06555
35	64.51875	47.57056	88	77.18364	45.63001	141	117.1052	45.05139
36	60.23732	47.42847	89	78.91327	45.57275	142	117.8061	45.08948
37	57.04492	47.31592	90	80.24351	45.53137	143	117.8538	45.04712
38	54.29257	47.17725	91	81.3653	45.46545	144	118.1431	45.06677
39	52.04154	47.04614	92	82.45414	45.39478	145	117.9223	45.0354
40	50.17103	46.79504	93	83.92077	45.30164	146	117.9905	45.04273
41	49.97886	46.63782	94	85.87289	45.26379	147	117.6423	45.04468
42	50.67817	46.39807	95	87.74128	45.22546	148	117.4543	44.995
43	52.50951	46.21631	96	89.45258	45.13904	149	117.8724	45.05579
44	54.80264	46.04443	97	91.63678	45.08203	150	117.5083	45.00903
45	57.61539	45.86316	98	93.96608	44.93018	151	117.3287	44.99353
46	61.20048	45.68811	99	97.22086	44.91382	152	117.3318	45.02283
47	65.31299	45.4801	100	99.83298	44.90369	153	116.7541	44.95288
48	70.10805	45.37122	101	102.0304	44.76392	154	117.6059	44.92737
49	74.49564	45.224	102	105.4044	44.72339	155	118.8744	44.87988
50	79.03646	45.03894	103	108.7518	44.64356	156	120.1917	44.88696
51	84.25724	44.85425	104	112.1199	44.68018	157	121.3529	44.93445
52	90.30894	44.72937	105	114.4308	44.61304	158	121.3953	44.92578
53	96.52834	44.51868	106	117.5885	44.58508	159	121.2097	44.92444
54	104.1773	44.42358	107	120.6803	44.51343	160	120.6521	44.91638
55	111.74	44.35901	108	124.2753	44.60291	161	120.1159	44.91907
56	118.5	44.27734	109	126.0038	44.54395	162	118.9661	44.94544
57	124.8577	44.30933	110	128.5637	44.57507	163	117.3133	44.94873
58	129.9496	44.25794	111	130.5545	44.49573	164	115.8796	44.92615
59	135.5138	44.29541	112	133.2702	44.5791	165	115.1788	44.8656
60	140.4408	44.38525	113	134.3662	44.60303	166	115.3198	44.9043
61	144.5337	44.41248	114	134.9786	44.68494	167	115.2844	44.91272
62	148.7853	44.59827	115	134.4432	44.72803	168	115.9	44.8971
63	150.4837	44.73584	116	133.7563	44.74475	169	117.179	44.88623
64	150.4572	44.89453	117	132.9855	44.78174	170	119.0659	44.88648
65	148.4314	45.13501	118	132.1898	44.84924	171	121.3491	44.89539
66	143.5124	45.25659	119	131.0397	44.85303	172	123.4457	44.86841
67	137.5506	45.3855	120	130.1317	44.91089	173	126.1914	44.89514
68	130.8465	45.57825	121	129.0826	44.89954	174	128.8388	44.92297
69	122.675	45.6471	122	128.9765	44.97571	175	130.5748	44.87732

176	132.3166	44.90308
177	132.772	44.90393
178	132.4979	44.90295
179	131.9243	44.8927
180	130.8647	44.86987
181	131.0611	44.88171
182	131.622	44.88513
183	132.4758	44.87366
184	133.6505	44.88611
185	134.6404	44.86084
186	135.9868	44.84009
187	137.4671	44.87073
188	138.2819	44.87512
189	138.942	44.83582
190	140.0966	44.87293
191	140.5742	44.82861
192	141.6282	44.83814
193	142.372	44.84375
194	142.8925	44.87598
195	142.826	44.89343
196	142.4936	44.88037
197	142.364	44.87256
198	142.2918	44.86511
199	142.2521	44.86206
200	142.1848	44.89673
201	141.5154	44.95667
202	140.0263	44.97852
203	138.4132	44.94836
204	137.4647	44.97412
205	136.2915	44.97754
206	135.251	45.03333
207	133.5999	45.05457
208	131.943	45.0625
209	130.4693	45.07007
210	129.1145	45.02722
211	128.5337	45.09522
212	127.0506	45.04932
213	126.4022	45.11353
214	124.952	45.10901
215	123.8034	45.11987
216	122.7055	45.07581
217	122.3695	45.15027
218	121.0737	45.13648
219	120.1392	45.19336
220	118.5637	45.26685
221	116.2357	45.26196
222	114.3736	45.25342
223	112.9257	45.26733
224	111.5613	45.26379
225	110.4901	45.24915
226	109.7925	45.24683
227	109.2557	45.1936
228	109.4733	45.22205
229	109.2665	45.23425
230	108.8839	45.22461
231	108.6362	45.23621
232	108.2407	45.1842
233	108.5143	45.26331
234	107.6744	45.22986
235	107.3295	45.24609
236	106.7459	45.18469
237	106.8894	45.20056
238	106.6516	45.25378
239	105.6121	45.23059
240	104.9203	45.24158
241	104.1191	45.25891
242	103.1947	45.24841
243	102.5301	45.24243
244	102.0282	45.23303
245	101.7178	45.21631
246	101.6438	45.23438
247	101.3254	45.23242
248	101.0294	45.18677
249	101.2702	45.19702
250	101.2518	45.21985
251	100.8763	45.20776
252	100.6393	45.21777
253	100.2265	45.17847
254	100.2756	45.2096
255	99.84906	45.19348
256	99.61833	45.14929
257	99.88965	45.23926
258	99.00185	45.20789
259	98.60952	45.19421
260	98.42835	45.17249
261	98.50187	45.18494
262	98.38732	45.12854
263	98.88954	45.13171
264	99.19848	45.14868
265	99.17806	45.09253
266	99.74821	45.13062
267	99.72251	45.11169
268	99.87679	45.08411
269	100.3163	45.06787
270	100.8725	45.11157
271	100.8112	45.14026
272	100.4296	45.11865
273	100.3818	45.18323
274	99.59987	45.12341
275	99.70906	45.11963
276	99.8782	45.11182
277	100.1282	45.11072
278	100.3675	45.08899
279	100.8233	45.08118
280	101.2656	45.06458
281	101.7818	45.10498
282	101.6499	45.08569
283	101.7226	45.11658
284	101.3931	45.07239
285	101.6517	45.07385
286	101.8495	45.07654
287	101.9878	45.04968
288	102.4173	45.0636
289	102.6143	45.06873
290	102.7175	45.09168
291	102.5459	45.07581
292	102.6135	45.09631
293	102.4268	45.07019
294	102.5761	45.09644
295	102.3689	45.03833
296	102.8682	45.09509
297	102.551	45.04712
298	102.8304	45.02942
299	103.2223	45.02722
300	103.4981	45.01868

6. Closed-loop response of the heat exchanger process for a regulatory DMC control, when a step disturbance of 100 cc/min is given in the cold water flow rate.

Time (s)	HWFR (cc/min)	Temp (oC)
0	76.77101	44.69739
1	76.79601	44.71675
2	76.82101	44.73611
3	76.84601	44.75548
4	76.87101	44.77484
5	76.89601	44.79421
6	76.92101	44.81357
7	76.94601	44.83293
8	76.97101	44.8523
9	77.08011	44.87166
10	77.18921	44.89103
11	77.2983	44.91039
12	77.4074	44.92976
13	77.5165	44.94912

14	77.6256	44.96848	69	80.95008	44.84682	124	102.9416	38.59535
15	77.7347	44.98785	70	80.96199	44.85287	125	103.8768	38.53088
16	77.8438	45.00721	71	80.9739	44.85892	126	104.8121	38.46641
17	77.99713	44.99895	72	80.98581	44.86497	127	105.7473	38.40194
18	78.15047	44.99069	73	81.01753	44.89802	128	106.6825	38.33747
19	78.30381	44.98243	74	81.04925	44.93107	129	107.4497	38.46388
20	78.45715	44.97417	75	81.08097	44.96412	130	108.2168	38.5903
21	78.61049	44.9659	76	81.11269	44.99716	131	108.984	38.71672
22	78.76383	44.95764	77	81.14441	45.03021	132	109.7511	38.84314
23	78.91717	44.94938	78	81.17613	45.06326	133	110.5183	38.96956
24	79.07051	44.94112	79	81.20785	45.09631	134	111.2854	39.09598
25	79.12799	44.94966	80	81.23957	45.12936	135	112.0526	39.2224
26	79.18547	44.95821	81	81.22315	45.10522	136	112.8198	39.34882
27	79.24294	44.96675	82	81.20673	45.08108	137	113.9568	39.41981
28	79.30042	44.9753	83	81.19031	45.05695	138	115.0938	39.49081
29	79.3579	44.98384	84	81.17389	45.03261	139	116.2308	39.56181
30	79.41537	44.99239	85	81.15747	45.00868	140	117.3678	39.63281
31	79.47285	45.00093	86	81.14105	44.98454	141	118.5048	39.70381
32	79.53033	45.00948	87	81.12463	44.9604	142	119.6418	39.7748
33	79.55985	45.02433	88	81.10821	44.93627	143	120.7788	39.8458
34	79.58938	45.03917	89	81.0689	44.96439	144	121.9159	39.9168
35	79.61891	45.05402	90	81.02958	44.99251	145	124.1889	39.97198
36	79.64843	45.06886	91	80.99027	45.02063	146	126.462	40.02715
37	79.67796	45.08371	92	80.95096	45.04876	147	128.7351	40.08233
38	79.70749	45.09856	93	80.91164	45.07688	148	131.0082	40.1375
39	79.73701	45.1134	94	80.87233	45.105	149	133.2813	40.19268
40	79.76654	45.12825	95	80.83302	45.13312	150	135.5544	40.24786
41	79.77122	45.0984	96	80.7937	45.16124	151	137.8275	40.30303
42	79.77591	45.06855	97	80.75311	45.08734	152	140.1006	40.35821
43	79.78059	45.0387	98	80.71252	45.01344	153	141.6155	40.54799
44	79.78527	45.00885	99	80.67192	44.93953	154	143.1304	40.73777
45	79.78995	44.979	100	80.63133	44.86563	155	144.6453	40.92755
46	79.79464	44.94915	101	80.59073	44.79173	156	146.1602	41.11732
47	79.79932	44.9193	102	80.55014	44.71782	157	147.6751	41.3071
48	79.804	44.88945	103	80.50955	44.64392	158	149.19	41.49688
49	79.86442	44.88849	104	80.46895	44.57002	159	150.7049	41.68666
50	79.92483	44.88753	105	81.56907	44.07811	160	152.2199	41.87644
51	79.98524	44.88658	106	82.66918	43.58619	161	153.705	42.00746
52	80.04565	44.88562	107	83.76929	43.09428	162	155.1902	42.13848
53	80.10607	44.88467	108	84.8694	42.60237	163	156.6754	42.26949
54	80.16648	44.88371	109	85.96951	42.11046	164	158.1606	42.40051
55	80.22689	44.88275	110	87.06962	41.61855	165	159.6457	42.53153
56	80.28731	44.8818	111	88.16973	41.12664	166	161.1309	42.66255
57	80.36271	44.87364	112	89.26985	40.63473	167	162.6161	42.79356
58	80.43811	44.86549	113	90.51119	40.41204	168	164.1013	42.92458
59	80.51351	44.85733	114	91.75253	40.18935	169	165.3133	42.9879
60	80.58892	44.84918	115	92.99388	39.96667	170	166.5254	43.05123
61	80.66432	44.84102	116	94.23522	39.74398	171	167.7374	43.11455
62	80.73972	44.83287	117	95.47657	39.52129	172	168.9495	43.17787
63	80.81513	44.82471	118	96.71791	39.2986	173	170.1615	43.24119
64	80.89053	44.81656	119	97.95925	39.07591	174	171.3736	43.30452
65	80.90244	44.82261	120	99.2006	38.85323	175	172.5856	43.36784
66	80.91435	44.82866	121	100.1358	38.78876	176	173.7977	43.43116
67	80.92626	44.83471	122	101.0711	38.72429	177	174.2272	43.5123
68	80.93817	44.84076	123	102.0063	38.65982	178	174.6568	43.59344

179	175.0863	43.67458	234	148.7911	45.51734	289	138.9065	43.85851
180	175.5159	43.75572	235	147.1629	45.51075	290	139.4247	43.88521
181	175.9455	43.83686	236	145.5346	45.50415	291	139.9429	43.91192
182	176.375	43.91801	237	143.9064	45.49756	292	140.4611	43.93862
183	176.8046	43.99915	238	142.2782	45.49097	293	140.9792	43.96532
184	177.2341	44.08029	239	140.6499	45.48437	294	141.4974	43.99203
185	177.4992	44.16106	240	139.0217	45.47778	295	142.0156	44.01873
186	177.7643	44.24183	241	137.8765	45.42771	296	142.5337	44.04543
187	178.0294	44.32261	242	136.7312	45.37763	297	142.7165	44.13074
188	178.2945	44.40338	243	135.586	45.32756	298	142.8992	44.21604
189	178.5597	44.48415	244	134.4407	45.27748	299	143.0819	44.30135
190	178.8248	44.56493	245	133.2955	45.22741	300	143.2646	44.38665
191	179.0899	44.6457	246	132.1503	45.17734	301	143.4474	44.47196
192	179.355	44.72647	247	131.005	45.12726	302	143.6301	44.55727
193	179.5393	44.76639	248	129.8598	45.07719	303	143.8128	44.64257
194	179.7235	44.80631	249	129.9111	44.99646	304	143.9955	44.72788
195	179.9078	44.84623	250	129.9625	44.91574	305	144.0898	44.74839
196	180.0921	44.88615	251	130.0139	44.83501	306	144.184	44.7689
197	180.2764	44.92607	252	130.0652	44.75428	307	144.2783	44.78941
198	180.4607	44.96599	253	130.1166	44.67356	308	144.3725	44.80993
199	180.645	45.00591	254	130.168	44.59283	309	144.4668	44.83044
200	180.8293	45.04583	255	130.2193	44.5121	310	144.5611	44.85095
201	180.1048	45.07956	256	130.2707	44.43138	311	144.6553	44.87146
202	179.3802	45.11329	257	130.3895	44.41662	312	144.7496	44.89197
203	178.6557	45.14701	258	130.5084	44.40186	313	144.7795	44.89986
204	177.9311	45.18074	259	130.6272	44.3871	314	144.8095	44.90774
205	177.2066	45.21447	260	130.7461	44.37234	315	144.8394	44.91562
206	176.482	45.24819	261	130.8649	44.35758	316	144.8694	44.92351
207	175.7575	45.28192	262	130.9838	44.34282	317	144.8993	44.93139
208	175.0329	45.31565	263	131.1026	44.32807	318	144.9293	44.93928
209	174.2697	45.32781	264	131.2215	44.31331	319	144.9592	44.94716
210	173.5066	45.33997	265	131.4539	44.29114	320	144.9891	44.95504
211	172.7434	45.35214	266	131.6864	44.26898	321	144.8925	44.99811
212	171.9802	45.3643	267	131.9189	44.24682	322	144.7958	45.04118
213	171.2171	45.37647	268	132.1514	44.22465	323	144.6991	45.08424
214	170.4539	45.38863	269	132.3839	44.20249	324	144.6024	45.12731
215	169.6907	45.4008	270	132.6164	44.18033	325	144.5057	45.17037
216	168.9275	45.41296	271	132.8489	44.15816	326	144.409	45.21344
217	167.9322	45.43605	272	133.0814	44.136	327	144.3123	45.2565
218	166.9368	45.45914	273	133.3344	44.10541	328	144.2156	45.29957
219	165.9414	45.48223	274	133.5875	44.07483	329	144.1093	45.28431
220	164.9461	45.50532	275	133.8406	44.04424	330	144.0029	45.26905
221	163.9507	45.52841	276	134.0937	44.01365	331	143.8965	45.25379
222	162.9553	45.55149	277	134.3468	43.98307	332	143.7901	45.23853
223	161.96	45.57458	278	134.5998	43.95248	333	143.6837	45.22327
224	160.9646	45.59767	279	134.8529	43.92189	334	143.5773	45.20801
225	159.85	45.58928	280	135.106	43.89131	335	143.4709	45.19275
226	158.7353	45.58088	281	135.5163	43.88387	336	143.3645	45.17749
227	157.6207	45.57249	282	135.9266	43.87643	337	143.3109	45.19205
228	156.5061	45.5641	283	136.3369	43.86899	338	143.2573	45.20662
229	155.3915	45.5557	284	136.7472	43.86156	339	143.2037	45.22118
230	154.2769	45.54731	285	137.1575	43.85412	340	143.1501	45.23575
231	153.1622	45.53892	286	137.5678	43.84668	341	143.0965	45.25031
232	152.0476	45.53052	287	137.9781	43.83925	342	143.0429	45.26488
233	150.4194	45.52393	288	138.3884	43.83181	343	142.9893	45.27944

344	142.9357	45.29401	399	135.4643	45.04364	454	137.4514	45.04821
345	142.8356	45.30641	400	135.4202	45.03799	455	137.4292	45.06185
346	142.7354	45.31882	401	135.4572	45.02676	456	137.4071	45.07548
347	142.6352	45.33123	402	135.4942	45.01554	457	137.3744	45.05918
348	142.535	45.34363	403	135.5312	45.00432	458	137.3418	45.04288
349	142.4349	45.35604	404	135.5682	44.9931	459	137.3091	45.02658
350	142.3347	45.36845	405	135.6052	44.98187	460	137.2764	45.01028
351	142.2345	45.38086	406	135.6421	44.97065	461	137.2438	44.99398
352	142.1343	45.39326	407	135.6791	44.95943	462	137.2111	44.97768
353	141.9662	45.3988	408	135.7161	44.9482	463	137.1785	44.96138
354	141.798	45.40433	409	135.74	44.94026	464	137.1458	44.94508
355	141.6298	45.40986	410	135.764	44.93232	465	137.1904	44.9403
356	141.4617	45.4154	411	135.7879	44.92437	466	137.2349	44.93552
357	141.2935	45.42093	412	135.8119	44.91643	467	137.2795	44.93074
358	141.1254	45.42646	413	135.8358	44.90849	468	137.324	44.92596
359	140.9572	45.432	414	135.8597	44.90054	469	137.3686	44.92117
360	140.789	45.43753	415	135.8837	44.8926	470	137.4131	44.91639
361	140.6681	45.43295	416	135.9076	44.88466	471	137.4577	44.91161
362	140.5472	45.42838	417	135.903	44.88548	472	137.5022	44.90683
363	140.4263	45.4238	418	135.8984	44.88631	473	137.5028	44.92847
364	140.3054	45.41922	419	135.8938	44.88714	474	137.5034	44.9501
365	140.1845	45.41464	420	135.8892	44.88797	475	137.504	44.97174
366	140.0635	45.41006	421	135.8845	44.8888	476	137.5046	44.99338
367	139.9426	45.40549	422	135.8799	44.88963	477	137.5052	45.01501
368	139.8217	45.40091	423	135.8753	44.89045	478	137.5058	45.03665
369	139.6658	45.3778	424	135.8707	44.89128	479	137.5064	45.05829
370	139.51	45.35469	425	135.8904	44.8858	480	137.507	45.07992
371	139.3541	45.33158	426	135.91	44.88032	481	137.5161	45.05838
372	139.1982	45.30847	427	135.9297	44.87484	482	137.5251	45.03683
373	139.0423	45.28536	428	135.9493	44.86936	483	137.5342	45.01528
374	138.8864	45.26225	429	135.969	44.86388	484	137.5432	44.99373
375	138.7306	45.23913	430	135.9886	44.8584	485	137.5523	44.97219
376	138.5747	45.21602	431	136.0083	44.85292	486	137.5613	44.95064
377	138.3852	45.20189	432	136.0279	44.84744	487	137.5704	44.92909
378	138.1956	45.18776	433	136.1009	44.8518	488	137.5795	44.90754
379	138.0061	45.17363	434	136.1739	44.85616	489	137.5194	44.92941
380	137.8166	45.1595	435	136.2468	44.86052	490	137.4594	44.95128
381	137.6271	45.14537	436	136.3198	44.86488	491	137.3994	44.97315
382	137.4376	45.13124	437	136.3927	44.86924	492	137.3394	44.99502
383	137.248	45.11711	438	136.4657	44.8736	493	137.2794	45.01689
384	137.0585	45.10298	439	136.5386	44.87796	494	137.2194	45.03876
385	136.8978	45.10051	440	136.6116	44.88232	495	137.1593	45.06063
386	136.7371	45.09804	441	136.7332	44.89283	496	137.0993	45.08249
387	136.5763	45.09557	442	136.8549	44.90334	497	136.9758	45.08734
388	136.4156	45.0931	443	136.9765	44.91385	498	136.8522	45.09219
389	136.2549	45.09063	444	137.0981	44.92436	499	136.7286	45.09703
390	136.0942	45.08815	445	137.2197	44.93488	500	136.6051	45.10188
391	135.9334	45.08568	446	137.3414	44.94539	501	136.4815	45.10672
392	135.7727	45.08321	447	137.463	44.9559	502	136.358	45.11157
393	135.7287	45.07756	448	137.5846	44.96641	503	136.2344	45.11642
394	135.6846	45.07191	449	137.5624	44.98004	504	136.1108	45.12126
395	135.6405	45.06625	450	137.5402	44.99368	505	136.0579	45.12163
396	135.5965	45.0606	451	137.518	45.00731	506	136.0049	45.122
397	135.5524	45.05495	452	137.4958	45.02095	507	135.9519	45.12237
398	135.5084	45.04929	453	137.4736	45.03458	508	135.8989	45.12274

509	135.8459	45.12311
510	135.7929	45.12348
511	135.7399	45.12385
512	135.6869	45.12422
513	135.6534	45.11464
514	135.6198	45.10506
515	135.5863	45.09548
516	135.5528	45.0859
517	135.5192	45.07632
518	135.4857	45.06674
519	135.4521	45.05716
520	135.4186	45.04758
521	135.3115	45.06193
522	135.2045	45.07628
523	135.0974	45.09063
524	134.9904	45.10499
525	134.8833	45.11934
526	134.7763	45.13369
527	134.6693	45.14804
528	134.5622	45.1624
529	134.5279	45.15245
530	134.4936	45.1425
531	134.4592	45.13256
532	134.4249	45.12261
533	134.3906	45.11267
534	134.3563	45.10272
535	134.3219	45.09278
536	134.2876	45.08283
537	134.1706	45.08932
538	134.0535	45.09581
539	133.9365	45.1023
540	133.8195	45.10878
541	133.7025	45.11527
542	133.5854	45.12176
543	133.4684	45.12825
544	133.3514	45.13474
545	133.2734	45.14094
546	133.1955	45.14714
547	133.1176	45.15333
548	133.0396	45.15953
549	132.9617	45.16573
550	132.8838	45.17193
551	132.8058	45.17813
552	132.7279	45.18433
553	132.6042	45.18848
554	132.4805	45.19263
555	132.3567	45.19678
556	132.233	45.20093
557	132.1093	45.20508
558	131.9856	45.20923
559	131.8619	45.21338
560	131.7381	45.21753
561	131.6508	45.20439
562	131.5634	45.19125
563	131.4761	45.1781
564	131.3887	45.16496
565	131.3014	45.15181
566	131.2141	45.13867
567	131.1267	45.12553
568	131.0394	45.11238
569	131.0709	45.10457
570	131.1025	45.09677
571	131.134	45.08896
572	131.1656	45.08115
573	131.1971	45.07334
574	131.2287	45.06553
575	131.2602	45.05773
576	131.2918	45.04992
577	131.2181	45.07354
578	131.1444	45.09716
579	131.0707	45.12078
580	130.997	45.1444
581	130.9233	45.16802
582	130.8496	45.19164
583	130.7759	45.21526
584	130.7022	45.23889
585	130.6343	45.26119
586	130.5664	45.2835
587	130.4986	45.3058
588	130.4307	45.32811
589	130.3628	45.35042
590	130.295	45.37272
591	130.2271	45.39503
592	130.1592	45.41734
593	130.155	45.38688
594	130.1507	45.35643
595	130.1465	45.32598
596	130.1422	45.29553
597	130.138	45.26508
598	130.1337	45.23462
599	130.1295	45.20417
600	130.1252	45.17372

7. Closed loop response of the heat exchanger process using WDMC for a step change of 10°C in the set point (35°C to 45°C).

Time (s)	HWFR (cc/min)	Temp (°C)
0	100.0123	35.20898
1	140.0118	35.33679
2	202.1457	35.46558
3	250.5842	35.7157
4	278.3495	36.07092
5	281.1595	36.55701
6	281.1595	37.30176
7	281.1595	38.09668
8	281.1595	38.7666
9	281.1595	39.40625
10	281.1595	40.10278
11	281.1595	40.99402
12	281.1595	41.64209
13	281.1595	42.1333
14	281.1595	42.73987
15	281.1595	43.25855
16	281.1595	43.73901
17	281.1595	44.25818
18	281.1595	44.78186
19	281.1595	45.23084
20	281.1595	45.58386
21	281.1595	45.82593
22	275.1162	46.0155
23	265.8315	46.01235
24	254.2845	45.99124
25	241.6576	45.97812
26	228.4459	45.92293
27	214.9458	45.92732
28	201.2446	45.85127
29	187.2091	45.78352
30	173.426	45.72786
31	159.8359	45.64656
32	146.9655	45.54731
33	134.7253	45.48079
34	126.548	45.44478
35	121.1744	45.35469
36	117.4019	45.24824
37	114.9244	45.22908
38	113.4544	45.14094
39	112.2961	45.08467
40	111.6539	44.98396
41	111.1557	44.91926
42	110.7518	44.85237
43	110.508	44.8321
44	110.5799	44.75618
45	110.3833	44.68379

46	109.7709	44.70076	99	106.4601	44.85796	152	134.429	45.12973
47	108.9994	44.63997	100	108.9898	44.85161	153	130.6204	45.23483
48	107.9165	44.67415	101	110.9375	44.86577	154	127.0068	45.21042
49	106.6362	44.63484	102	112.4243	44.81853	155	123.8082	45.22982
50	105.0966	44.62568	103	113.9162	44.82744	156	120.7153	45.18783
51	102.6266	44.65205	104	114.8627	44.85503	157	117.5187	45.17843
52	100.7101	44.65461	105	115.6855	44.84966	158	114.6635	45.17074
53	98.42621	44.7082	106	116.1533	44.87773	159	111.8731	45.14621
54	96.07038	44.69587	107	116.2532	44.85588	160	109.1351	45.06308
55	93.2805	44.75691	108	116.1725	44.87139	161	106.2429	45.04013
56	90.48587	44.78267	109	115.9004	44.86138	162	103.7683	45.02548
57	87.52457	44.81111	110	115.5495	44.86028	163	101.2696	44.97165
58	84.66937	44.91304	111	114.7839	44.8593	164	99.14428	44.93002
59	81.66755	44.96321	112	114.2954	44.87712	165	97.22112	44.93931
60	79.0438	45.00386	113	113.8632	44.90923	166	95.2573	44.94023
61	76.41835	45.04182	114	113.2432	44.90606	167	93.52452	44.96799
62	73.81426	45.13228	115	113.2362	44.95122	168	91.68648	44.97965
63	71.22763	45.17073	116	113.253	44.96111	169	90.34552	44.98623
64	68.79534	45.25081	117	113.4221	44.9854	170	88.97994	45.038
65	66.63085	45.25119	118	113.6518	44.97856	171	87.91076	44.99796
66	64.5521	45.24191	119	113.9569	44.93779	172	87.00902	44.99161
67	62.58511	45.21261	120	114.6677	44.97002	173	86.18201	45.00577
68	61.06343	45.24398	121	115.2509	44.94219	174	85.53316	44.95853
69	59.87917	45.2015	122	116.1609	44.94951	175	84.64449	44.96744
70	58.82078	45.20273	123	117.0294	44.971	176	83.97626	44.99503
71	58.12518	45.15524	124	118.0777	44.96709	177	83.22028	44.98966
72	57.64545	45.12289	125	119.2553	44.96404	178	82.68179	45.01773
73	57.51174	45.14535	126	120.6116	44.91167	179	82.21994	44.99588
74	57.5635	45.10592	127	122.309	44.96538	180	81.47674	45.01139
75	57.90201	45.07003	128	124.3873	44.93584	181	80.84499	45.00138
76	58.38415	45.05734	129	126.7769	44.93588	182	80.29153	45.00028
77	59.16172	45.02084	130	129.6253	44.98788	183	79.67057	44.9993
78	60.0315	45.03256	131	132.6206	44.98983	184	79.13153	45.01712
79	61.21305	45.02548	132	135.6302	44.9913	185	78.37468	45.04923
80	62.38341	45.01156	133	138.4805	44.97372	186	77.54223	45.04606
81	63.85415	45.00155	134	141.43	45.0123	187	76.69757	45.09122
82	65.41499	44.96078	135	144.6205	45.03231	188	75.83905	45.10111
83	67.13265	44.9797	136	147.5797	45.0903	189	75.15241	45.1254
84	69.03069	44.96627	137	150.2664	45.08334	190	74.26818	45.11856
85	70.93051	44.95041	138	152.8584	45.07577	191	73.41109	45.07779
86	73.02744	44.94906	139	155.086	45.12204	192	72.59608	45.11002
87	75.37645	44.91891	140	156.7718	45.09799	193	71.73723	45.08219
88	77.71913	44.94772	141	157.9878	45.06198	194	70.99577	45.08951
89	80.34826	44.93051	142	158.7367	45.07736	195	70.04188	45.111
90	82.75322	44.87606	143	158.4104	45.08297	196	69.15743	45.10709
91	85.08899	44.85702	144	157.579	45.09103	197	68.55244	45.10404
92	87.6202	44.9161	145	156.3459	45.11496	198	68.00352	45.05167
93	90.16751	44.86569	146	154.2963	45.08249	199	67.07834	45.10538
94	92.85194	44.84286	147	151.7941	45.11691	200	66.41807	45.07584
95	95.37344	44.83273	148	148.5338	45.14023	201	65.84005	45.05007
96	98.21278	44.83965	149	145.152	45.08993	202	65.28581	45.06935
97	101.0785	44.78862	150	141.7316	45.13827	203	64.84869	45.1227
98	103.8986	44.898	151	138.1454	45.1307	204	64.46588	45.09365

205	63.99127	45.06935
206	63.55726	45.10134
207	63.31461	45.084
208	63.20796	45.10256
209	63.1269	45.08596
210	63.16526	45.07961
211	63.23578	45.11061
212	63.75146	45.12746
213	64.11466	45.11562
214	64.76639	45.09133
215	65.64305	45.08766
216	66.66088	45.09316
217	67.95549	45.0873
218	69.59923	45.00453
219	71.38677	45.00905
220	73.13918	45.03139
221	75.1834	45.04127
222	77.61835	45.03224
223	79.89501	44.99696
224	82.46103	44.99586
225	85.11428	45.01015
226	87.40263	44.99452
227	89.67967	44.98903
228	92.00952	44.97926
229	94.43609	44.9983
230	97.12001	45.00282
231	100.069	45.00465
232	102.9236	45.00258
233	106.1098	45.02235
234	109.5888	45.03798
235	112.692	45.04188
236	116.0402	45.04371
237	118.9163	45.06349
238	121.9249	45.10035
239	124.8414	45.11659
240	127.4214	45.1233
241	129.8975	45.16084
242	131.8909	45.17256
243	133.9821	45.16548
244	135.5886	45.15156
245	137.0709	45.14155
246	138.4077	45.10078
247	139.0466	45.1197
248	139.4903	45.10627
249	139.8403	45.09041
250	140.142	45.08906
251	139.9956	45.05891
252	139.8613	45.08772
253	139.393	45.07051
254	138.7424	45.01606
255	138.1447	44.99702
256	137.148	45.0561
257	135.882	45.00569
258	135.0001	44.98286
259	133.7553	44.97273
260	131.904	44.97965
261	130.4213	44.92862
262	128.5497	45.038
263	126.4849	44.99796
264	124.6389	44.99161
265	122.7422	45.00577
266	121.0741	44.95853
267	119.3363	44.96744
268	117.7718	44.99503
269	116.307	44.98966
270	115.0736	45.01773
271	114.0803	44.99588
272	112.9076	45.01139
273	112.284	44.91577
274	110.9626	44.86853
275	110.0905	44.87744
276	109.375	44.90503
277	108.6247	44.89966
278	108.281	44.92773
279	107.884	44.90588
280	107.2767	44.92139
281	106.7123	44.91138
282	105.9485	44.91028
283	105.3647	44.9093
284	104.663	44.92712
285	104.042	44.95923
286	103.4195	44.95606
287	102.7911	45.00122
288	102.0299	45.01111
289	101.0357	45.0354
290	100.0769	45.02856
291	98.83791	44.98779
292	97.58326	45.02002
293	96.19818	44.99219
294	94.9251	44.99951
295	93.95772	45.021
296	92.76263	45.01709
297	91.76735	45.01404
298	90.71143	45.07042
299	89.51865	45.06892
300	88.36611	45.02839

8. Closed loop response of the heat exchanger process using WDMC for a step change of 100 cc/min in the cold water flow rate.

Time (s)	HWRF (cc/min)	Temp (^{o}C)
0	73.38079	44.90231
1	73.39781	44.90492
2	73.41484	44.90753
3	73.43187	44.91014
4	73.44889	44.91275
5	73.46592	44.91536
6	73.48295	44.91797
7	73.49997	44.92058
8	73.517	44.92319
9	73.53403	44.92579
10	73.55105	44.9284
11	73.56808	44.93101
12	73.58511	44.93362
13	73.60213	44.93623
14	73.61916	44.93884
15	73.63619	44.94145
16	73.65321	44.94406
17	73.67024	44.94858
18	73.68727	44.9531
19	73.70429	44.95762
20	73.72132	44.96214
21	73.73835	44.96666
22	73.75537	44.97117
23	73.7724	44.97569
24	73.78943	44.98021
25	73.80645	44.98473
26	73.82348	44.98925
27	73.84051	44.99377
28	73.85753	44.99829
29	73.87456	45.00281
30	73.89159	45.00733
31	73.90861	45.01185
32	73.92564	45.01637
33	73.94266	45.01391
34	73.95969	45.01144
35	73.97672	45.00897
36	73.99374	45.0065
37	74.01077	45.00403
38	74.0278	45.00157
39	74.04482	44.9991
40	74.06185	44.99663
41	74.07888	44.99416
42	74.0959	44.99169
43	74.11293	44.98923

44	74.12996	44.98676	99	76.96275	44.71516	154	127.7785	41.33622
45	74.14698	44.98429	100	77.61189	44.6195	155	128.8086	41.38983
46	74.16401	44.98182	101	78.26103	44.52385	156	129.8387	41.44345
47	74.18104	44.97936	102	78.91016	44.42819	157	130.8688	41.49706
48	74.19806	44.97689	103	79.5593	44.33254	158	131.8989	41.55067
49	74.21509	44.97728	104	80.20844	44.23688	159	132.929	41.60428
50	74.23212	44.97767	105	80.85757	44.14123	160	133.9591	41.65789
51	74.24914	44.97806	106	81.50671	44.04557	161	135.2631	41.75849
52	74.26617	44.97846	107	82.15585	43.94992	162	136.5671	41.85909
53	74.2832	44.97885	108	82.80499	43.85426	163	137.8711	41.95969
54	74.30022	44.97924	109	83.45412	43.75861	164	139.1751	42.06029
55	74.31725	44.97963	110	84.10326	43.66295	165	140.4791	42.16089
56	74.33428	44.98002	111	84.7524	43.5673	166	141.7831	42.2615
57	74.3513	44.98042	112	85.40153	43.47165	167	143.0871	42.3621
58	74.36833	44.98081	113	86.09847	43.34679	168	144.3912	42.4627
59	74.38536	44.9812	114	86.79541	43.22194	169	145.6952	42.5633
60	74.40238	44.98159	115	87.49235	43.09708	170	146.9992	42.6639
61	74.41941	44.98198	116	88.18929	42.97223	171	148.3032	42.7645
62	74.43644	44.98238	117	88.88622	42.84737	172	149.6072	42.8651
63	74.45346	44.98277	118	89.58316	42.72252	173	150.9112	42.96571
64	74.47049	44.98316	119	90.2801	42.59767	174	152.2152	43.06631
65	74.48752	44.98447	120	90.97704	42.47281	175	153.5192	43.16691
66	74.50454	44.98579	121	91.67398	42.34796	176	154.8233	43.26751
67	74.52157	44.9871	122	92.37092	42.2231	177	155.4341	43.34675
68	74.5386	44.98842	123	93.06785	42.09825	178	156.0449	43.42598
69	74.55562	44.98973	124	93.76479	41.97339	179	156.6557	43.50522
70	74.57265	44.99105	125	94.46173	41.84854	180	157.2665	43.58446
71	74.58968	44.99236	126	95.15867	41.72369	181	157.8774	43.6637
72	74.6067	44.99367	127	95.85561	41.59883	182	158.4882	43.74294
73	74.62373	44.99499	128	96.55254	41.47398	183	159.099	43.82217
74	74.64076	44.9963	129	97.86036	41.43186	184	159.7098	43.90141
75	74.65778	44.99762	130	99.16818	41.38974	185	160.3206	43.98065
76	74.67481	44.99893	131	100.476	41.34763	186	160.9315	44.05989
77	74.69183	45.00025	132	101.7838	41.30551	187	161.5423	44.13912
78	74.70886	45.00156	133	103.0916	41.2634	188	162.1531	44.21836
79	74.72589	45.00287	134	104.3994	41.22128	189	162.7639	44.2976
80	74.74291	45.00419	135	105.7073	41.17916	190	163.3747	44.37684
81	74.75994	45.00406	136	107.0151	41.13705	191	163.9856	44.45607
82	74.77697	45.00393	137	108.3229	41.09493	192	164.5964	44.53531
83	74.79399	45.0038	138	109.6307	41.05282	193	164.8758	44.61561
84	74.81102	45.00367	139	110.9385	41.0107	194	165.1552	44.69591
85	74.82805	45.00354	140	112.2463	40.96858	195	165.4346	44.77621
86	74.84507	45.00341	141	113.5542	40.92647	196	165.714	44.8565
87	74.8621	45.00328	142	114.862	40.88435	197	165.9934	44.9368
88	74.87913	45.00315	143	116.1698	40.84223	198	166.2728	45.0171
89	74.89615	45.00302	144	117.4776	40.80012	199	166.5522	45.0974
90	74.91318	45.0029	145	118.5077	40.85373	200	166.8316	45.17769
91	74.93021	45.00277	146	119.5378	40.90734	201	167.111	45.25799
92	74.94723	45.00264	147	120.5679	40.96095	202	167.3904	45.33829
93	74.96426	45.00251	148	121.598	41.01456	203	167.6698	45.41859
94	74.98129	45.00238	149	122.6281	41.06817	204	167.9492	45.49889
95	74.99831	45.00225	150	123.6582	41.12178	205	168.2286	45.57918
96	75.01534	45.00212	151	124.6882	41.17539	206	168.508	45.65948
97	75.66448	44.90647	152	125.7183	41.229	207	168.7874	45.73978
98	76.31361	44.81081	153	126.7484	41.28261	208	169.0668	45.82008

209	168.0721	45.76928	264	137.2131	44.92314	319	143.4896	44.52343
210	167.0773	45.71849	265	137.287	44.94339	320	143.3552	44.52119
211	166.0826	45.6677	266	137.361	44.96363	321	143.1811	44.53234
212	165.0878	45.61691	267	137.435	44.98388	322	143.0069	44.5435
213	164.0931	45.56612	268	137.5089	45.00413	323	142.8327	44.55466
214	163.0983	45.51532	269	137.5829	45.02438	324	142.6585	44.56582
215	162.1036	45.46453	270	137.6569	45.04463	325	142.4843	44.57697
216	161.1088	45.41374	271	137.7308	45.06488	326	142.3102	44.58813
217	160.1141	45.36295	272	137.8048	45.08513	327	142.136	44.59929
218	159.1193	45.31215	273	138.0112	45.08254	328	141.9618	44.61044
219	158.1246	45.26136	274	138.2175	45.07996	329	141.7876	44.6216
220	157.1298	45.21057	275	138.4239	45.07737	330	141.6135	44.63276
221	156.135	45.15978	276	138.6303	45.07478	331	141.4393	44.64392
222	155.1403	45.10899	277	138.8367	45.07219	332	141.2651	44.65507
223	154.1455	45.05819	278	139.043	45.0696	333	141.0909	44.66623
224	153.1508	45.0074	279	139.2494	45.06701	334	140.9167	44.67739
225	152.492	44.98085	280	139.4558	45.06443	335	140.7426	44.68854
226	151.8332	44.95429	281	139.6621	45.06184	336	140.5684	44.6997
227	151.1744	44.92774	282	139.8685	45.05925	337	140.173	44.70517
228	150.5156	44.90118	283	140.0749	45.05666	338	139.7777	44.71064
229	149.8568	44.87463	284	140.2812	45.05407	339	139.3823	44.71611
230	149.1979	44.84808	285	140.4876	45.05149	340	138.987	44.72158
231	148.5391	44.82152	286	140.694	45.0489	341	138.5916	44.72705
232	147.8803	44.79497	287	140.9003	45.04631	342	138.1963	44.73252
233	147.2215	44.76841	288	141.1067	45.04372	343	137.8009	44.73798
234	146.5627	44.74186	289	141.3816	45.01331	344	137.4056	44.74345
235	145.9039	44.7153	290	141.6566	44.9829	345	137.0102	44.74892
236	145.2451	44.68875	291	141.9315	44.95249	346	136.6149	44.75439
237	144.5863	44.6622	292	142.2064	44.92207	347	136.2195	44.75986
238	143.9275	44.63564	293	142.4814	44.89166	348	135.8242	44.76533
239	143.2687	44.60909	294	142.7563	44.86125	349	135.4288	44.7708
240	142.6099	44.58253	295	143.0312	44.83084	350	135.0335	44.77627
241	142.2356	44.5937	296	143.3061	44.80043	351	134.6381	44.78174
242	141.8613	44.60486	297	143.5811	44.77002	352	134.2428	44.78721
243	141.487	44.61602	298	143.856	44.7396	353	134.2135	44.79555
244	141.1127	44.62719	299	144.1309	44.70919	354	134.1843	44.8039
245	140.7384	44.63835	300	144.4059	44.67878	355	134.1551	44.81224
246	140.3642	44.64951	301	144.6808	44.64837	356	134.1259	44.82059
247	139.9899	44.66067	302	144.9557	44.61796	357	134.0967	44.82893
248	139.6156	44.67184	303	145.2306	44.58755	358	134.0675	44.83728
249	139.2413	44.683	304	145.5056	44.55714	359	134.0383	44.84562
250	138.867	44.69416	305	145.3712	44.55489	360	134.0091	44.85397
251	138.4927	44.70532	306	145.2368	44.55264	361	133.9799	44.86231
252	138.1184	44.71649	307	145.1024	44.55039	362	133.9507	44.87066
253	137.7442	44.72765	308	144.968	44.54815	363	133.9215	44.87901
254	137.3699	44.73881	309	144.8336	44.5459	364	133.8923	44.88735
255	136.9956	44.74998	310	144.6992	44.54365	365	133.863	44.8957
256	136.6213	44.76114	311	144.5648	44.54141	366	133.8338	44.90404
257	136.6953	44.78139	312	144.4304	44.53916	367	133.8046	44.91239
258	136.7692	44.80164	313	144.296	44.53691	368	133.7754	44.92073
259	136.8432	44.82189	314	144.1616	44.53467	369	133.9205	44.92376
260	136.9172	44.84214	315	144.0272	44.53242	370	134.0655	44.92679
261	136.9911	44.86239	316	143.8928	44.53017	371	134.2106	44.92982
262	137.0651	44.88264	317	143.7584	44.52793	372	134.3556	44.93285
263	137.1391	44.90289	318	143.624	44.52568	373	134.5006	44.93587

374	134.6457	44.9389	429	139.4813	45.08915	484	141.4965	45.10705
375	134.7907	44.94193	430	139.5206	45.09391	485	141.5345	45.10836
376	134.9358	44.94496	431	139.56	45.09867	486	141.5725	45.10968
377	135.0808	44.94799	432	139.5994	45.10343	487	141.6105	45.111
378	135.2259	44.95101	433	139.619	45.10635	488	141.6484	45.11232
379	135.3709	44.95404	434	139.6387	45.10927	489	141.6864	45.11364
380	135.5159	44.95707	435	139.6584	45.1122	490	141.7244	45.11496
381	135.661	44.9601	436	139.678	45.11512	491	141.7624	45.11628
382	135.806	44.96313	437	139.6977	45.11804	492	141.8004	45.1176
383	135.9511	44.96615	438	139.7174	45.12096	493	141.8384	45.11891
384	136.0961	44.96918	439	139.7371	45.12388	494	141.8763	45.12023
385	136.1672	44.97069	440	139.7567	45.1268	495	141.9143	45.12155
386	136.2383	44.97221	441	139.7764	45.12973	496	141.9523	45.12287
387	136.3094	44.97372	442	139.7961	45.13265	497	141.8958	45.12169
388	136.3805	44.97523	443	139.8157	45.13557	498	141.8394	45.12051
389	136.4516	44.97675	444	139.8354	45.13849	499	141.7829	45.11933
390	136.5227	44.97826	445	139.8551	45.14141	500	141.7264	45.11815
391	136.5938	44.97977	446	139.8748	45.14433	501	141.6699	45.11697
392	136.6649	44.98129	447	139.8944	45.14726	502	141.6134	45.11578
393	136.736	44.9828	448	139.9141	45.15018	503	141.557	45.1146
394	136.8071	44.98431	449	139.9416	45.14583	504	141.5005	45.11342
395	136.8782	44.98583	450	139.9691	45.14149	505	141.444	45.11224
396	136.9493	44.98734	451	139.9967	45.13714	506	141.3875	45.11106
397	137.0204	44.98885	452	140.0242	45.1328	507	141.3311	45.10988
398	137.0915	44.99037	453	140.0517	45.12846	508	141.2746	45.1087
399	137.1626	44.99188	454	140.0792	45.12411	509	141.2181	45.10752
400	137.2337	44.99339	455	140.1068	45.11977	510	141.1616	45.10633
401	137.3422	44.99551	456	140.1343	45.11542	511	141.1052	45.10515
402	137.4506	44.99763	457	140.1618	45.11108	512	141.0487	45.10397
403	137.5591	44.99974	458	140.1893	45.10673	513	141.0877	45.10779
404	137.6676	45.00186	459	140.2169	45.10239	514	141.1267	45.11161
405	137.7761	45.00398	460	140.2444	45.09805	515	141.1658	45.11543
406	137.8846	45.0061	461	140.2719	45.0937	516	141.2048	45.11925
407	137.9931	45.00821	462	140.2994	45.08936	517	141.2438	45.12307
408	138.1016	45.01033	463	140.327	45.08501	518	141.2828	45.12688
409	138.2101	45.01245	464	140.3545	45.08067	519	141.3219	45.1307
410	138.3186	45.01456	465	140.4164	45.08199	520	141.3609	45.13452
411	138.4271	45.01668	466	140.4782	45.08331	521	141.3999	45.13834
412	138.5356	45.0188	467	140.5401	45.08463	522	141.4389	45.14216
413	138.6441	45.02092	468	140.602	45.08594	523	141.478	45.14598
414	138.7525	45.02303	469	140.6639	45.08726	524	141.517	45.1498
415	138.861	45.02515	470	140.7258	45.08858	525	141.556	45.15362
416	138.9695	45.02727	471	140.7877	45.0899	526	141.595	45.15744
417	139.0089	45.03203	472	140.8495	45.09122	527	141.6341	45.16125
418	139.0483	45.03679	473	140.9114	45.09254	528	141.6731	45.16507
419	139.0876	45.04155	474	140.9733	45.09386	529	141.6345	45.16014
420	139.127	45.04631	475	141.0352	45.09518	530	141.5959	45.15521
421	139.1663	45.05107	476	141.0971	45.09649	531	141.5573	45.15028
422	139.2057	45.05583	477	141.1589	45.09781	532	141.5188	45.14535
423	139.2451	45.06059	478	141.2208	45.09913	533	141.4802	45.14042
424	139.2844	45.06535	479	141.2827	45.10045	534	141.4416	45.13549
425	139.3238	45.07011	480	141.3446	45.10177	535	141.403	45.13055
426	139.3632	45.07487	481	141.3826	45.10309	536	141.3645	45.12562
427	139.4025	45.07963	482	141.4206	45.10441	537	141.3259	45.12069
428	139.4419	45.08439	483	141.4585	45.10573	538	141.2873	45.11576

539	141.2487	45.11083	560	142.4421	45.07275	581	141.7709	45.07286
540	141.2102	45.1059	561	142.4278	45.07276	582	141.6826	45.07287
541	141.1716	45.10097	562	142.4134	45.07276	583	141.5942	45.07288
542	141.133	45.09604	563	142.3991	45.07277	584	141.5058	45.07288
543	141.0944	45.09111	564	142.3848	45.07277	585	141.4174	45.07289
544	141.0559	45.08617	565	142.3704	45.07278	586	141.3291	45.07289
545	141.1425	45.08534	566	142.3561	45.07278	587	141.2407	45.0729
546	141.2291	45.0845	567	142.3418	45.07279	588	141.1523	45.0729
547	141.3158	45.08366	568	142.3274	45.07279	589	141.064	45.07291
548	141.4024	45.08282	569	142.3131	45.0728	590	140.9756	45.07291
549	141.4891	45.08198	570	142.2988	45.0728	591	140.8872	45.07292
550	141.5757	45.08114	571	142.2844	45.07281	592	140.7989	45.07293
551	141.6623	45.0803	572	142.2701	45.07282	593	140.8331	45.07294
552	141.749	45.07946	573	142.2558	45.07282	594	140.8674	45.07295
553	141.8356	45.07862	574	142.2414	45.07283	595	140.9016	45.07296
554	141.9223	45.07778	575	142.2271	45.07283	596	140.9359	45.07297
555	142.0089	45.07695	576	142.2128	45.07284	597	140.9702	45.07298
556	142.0955	45.07611	577	142.1244	45.07284	598	141.0044	45.07299
557	142.1822	45.07527	578	142.036	45.07285	599	141.0387	45.073
558	142.2688	45.07443	579	141.9477	45.07285	600	141.0729	45.07301
559	142.3555	45.07359	580	141.8593	45.07286			

9. Open-loop response (process reaction curve) of the liquid level process for a step change of 50 cc/min in the inlet flow rate.

Time (s)	Level (mm)						
0	0	26	3.484054	53	6.225299	80	8.519509
1	0.030234	27	3.734329	54	6.346848	81	8.639395
2	0.081035	28	3.815366	55	6.540089	82	8.696912
3	0.179128	29	3.966068	56	6.609498	83	8.761581
4	0.249121	30	3.931811	57	6.72319	84	8.758181
5	0.376922	31	4.182595	58	6.793226	85	8.817091
6	0.524776	32	4.113614	59	6.751561	86	8.919056
7	0.619488	33	4.353584	60	6.846399	87	8.948531
8	0.721152	34	4.48452	61	7.055785	88	8.945617
9	0.872649	35	4.623354	62	7.113685	89	8.989338
10	1.239616	36	4.628764	63	7.106168	90	9.20568
11	1.255033	37	4.701204	64	7.234687	91	9.244828
12	1.303624	38	4.895892	65	7.345725	92	9.32035
13	1.506476	39	4.919871	66	7.458342	93	9.35383
14	1.701856	40	5.107475	67	7.507109	94	9.329703
15	1.843922	41	5.069033	68	7.511606	95	9.539451
16	2.215812	42	5.194733	69	7.747997	96	9.497767
17	2.295043	43	5.267811	70	7.769173	97	9.522923
18	2.334646	44	5.425701	71	7.81467	98	9.584051
19	2.610727	45	5.549559	72	7.944663	99	9.669198
20	2.776268	46	5.651608	73	8.022168	100	9.726398
21	2.901697	47	5.720374	74	7.988493	101	9.779831
22	3.019856	48	5.938294	75	8.045962	102	9.938107
23	3.251084	49	5.843807	76	8.257359	103	9.974239
24	3.281405	50	5.975489	77	8.179934	104	9.946137
25	3.391514	51	6.023156	78	8.241154	105	10.12894
		52	6.299351	79	8.453183	106	10.01174

107	10.20098	156	12.22653	205	13.72107	254	14.57262
108	10.20467	157	12.39647	206	13.73635	255	14.60795
109	10.27151	158	12.48098	207	13.80128	256	14.57734
110	10.27323	159	12.33785	208	13.71431	257	14.68331
111	10.33344	160	12.46682	209	13.83588	258	14.59441
112	10.48837	161	12.57914	210	13.82828	259	14.74391
113	10.53803	162	12.62078	211	13.84113	260	14.71354
114	10.56721	163	12.58168	212	13.9144	261	14.71836
115	10.6344	164	12.60695	213	13.9665	262	14.61832
116	10.72388	165	12.72136	214	13.81324	263	14.76036
117	10.70843	166	12.65633	215	13.8362	264	14.79683
118	10.77888	167	12.64256	216	13.97709	265	14.67086
119	10.87753	168	12.78643	217	14.03122	266	14.71342
120	10.80663	169	12.75318	218	13.91235	267	14.66768
121	10.95154	170	12.85494	219	13.89715	268	14.84449
122	10.94095	171	12.84899	220	13.95028	269	14.84483
123	11.05182	172	12.93103	221	14.04861	270	14.80127
124	11.01263	173	12.86599	222	14.08487	271	14.70159
125	11.01475	174	12.83897	223	14.003	272	14.84942
126	11.18936	175	12.84718	224	14.02826	273	14.86275
127	11.11954	176	12.865	225	14.14334	274	14.912
128	11.24963	177	13.00109	226	14.13581	275	14.77985
129	11.29552	178	13.04503	227	14.10324	276	14.93482
130	11.27388	179	13.04846	228	14.17051	277	14.82648
131	11.3089	180	13.09388	229	14.11485	278	14.82673
132	11.304	181	13.12622	230	14.13496	279	14.80011
133	11.52562	182	13.21945	231	14.2859	280	14.98586
134	11.43039	183	13.16758	232	14.32242	281	14.84726
135	11.52279	184	13.11428	233	14.20016	282	15.01254
136	11.53965	185	13.13116	234	14.25341	283	14.8599
137	11.69957	186	13.25648	235	14.29293	284	14.91457
138	11.72045	187	13.33349	236	14.39418	285	14.87604
139	11.74476	188	13.20689	237	14.36209	286	15.0563
140	11.80503	189	13.27381	238	14.42391	287	14.98022
141	11.77855	190	13.38518	239	14.36404	288	15.01416
142	11.84483	191	13.35347	240	14.45424	289	15.06562
143	11.8235	192	13.3689	241	14.33407	290	15.10354
144	11.80465	193	13.40772	242	14.33132	291	15.01387
145	12.00079	194	13.41142	243	14.49036	292	14.959
146	11.9371	195	13.41209	244	14.49333	293	14.99547
147	11.91715	196	13.54221	245	14.35293	294	15.10868
148	12.09132	197	13.52282	246	14.53947	295	15.11974
149	12.12996	198	13.62128	247	14.45909	296	15.10669
150	12.14351	199	13.58618	248	14.41555	297	14.99088
151	12.17851	200	13.59342	249	14.41077	298	15.00285
152	12.13526	201	13.62996	250	14.46039	299	15.08449
153	12.2727	202	13.65192	251	14.57042	300	15.05811
154	12.29549	203	13.65706	252	14.61257		
155	12.36262	204	13.60227	253	14.55721		

10. Closed loop response of the liquid level process for a step change of 30 mm in setpoint using PID controller.

Time (s)	FR (lit/min)	Level (mm)
0	2.100234	80.0543
1	2.100234	80.60254
2	2.532465	81.92822
3	2.905673	84.42725
4	3.019798	86.7334
5	3.017984	88.45947
6	3.016418	89.73877
7	3.014959	90.84229
8	3.01496	92.10449
9	3.015767	93.37891
10	3.012364	94.87549
11	3.008182	95.91553
12	3.012019	96.61865
13	3.028232	98.44482
14	3.004771	99.45801
15	3.008044	100.8569
16	3.006703	101.8652
17	3.001231	102.7979
18	3.010792	103.9111
19	2.995244	104.917
20	2.985588	105.9888
21	2.979083	107.3706
22	2.974958	107.9199
23	2.972193	108.96
24	2.958086	109.5581
25	2.932779	109.6411
26	2.907387	110.3149
27	2.60324	111.604
28	2.493297	111.1768
29	2.468867	111.8408
30	2.213717	112.7637
31	2.088209	112.793
32	1.712664	112.0288
33	1.443506	113.689
34	1.195833	113.6475
35	0.751758	112.8662
36	0.399315	112.6563
37	0.353263	112.7417
38	0.300435	112.5439
39	0.271519	111.9458
40	0.267971	111.4429
41	0.26093	111.5039
42	0.253736	111.2671
43	0.180399	111.4624
44	0.193164	111.1719
45	0.191493	110.6396
46	0.190371	110.874
47	0.194951	109.9658
48	0.196956	110.0122
49	0.18552	109.5581
50	0.184143	109.6338
51	0.198114	109.6948
52	0.183506	109.292
53	0.187713	109.4849
54	0.157642	109.1992
55	0.189584	108.584
56	0.1897	109.0063
57	0.188659	108.2275
58	0.184741	108.418
59	0.19942	109.187
60	0.188531	108.8794
61	0.194044	108.6499
62	0.195694	109.0723
63	0.200311	109.5532
64	0.194122	109.7559
65	0.201498	110.1172
66	0.208435	109.1943
67	0.210967	109.4336
68	0.218363	110.0366
69	0.229916	109.8682
70	0.232086	109.4751
71	0.239004	109.9951
72	0.239772	110.3711
73	0.276522	110.0854
74	0.277172	109.7803
75	0.277512	109.8291
76	0.293864	110.9033
77	0.310396	110.5933
78	0.318358	109.4678
79	0.318821	110.9155
80	0.319219	110.2905
81	0.317389	109.917
82	0.322451	109.9658
83	0.323115	109.9951
84	0.321838	109.8047
85	0.331919	110.0708
86	0.332786	110.4004
87	0.336786	110.6494
88	0.335431	110.2231
89	0.332328	110.4917
90	0.335783	110.4404
91	0.337703	110.6187
92	0.330348	110.4258
93	0.326721	110.4795
94	0.327494	110.2402
95	0.32223	110.3916
96	0.321453	110.123
97	0.321662	109.7104
98	0.327883	109.8691
99	0.335373	109.8887
100	0.338472	110.2134
101	0.33715	110.8896
102	0.330267	110.8066
103	0.32758	110.1768
104	0.316367	110.4429
105	0.302466	110.7285
106	0.29874	110.2305
107	0.287777	110.4697
108	0.280187	111.1191
109	0.276861	111.6733
110	0.276205	111.3364
111	0.271914	110.8799
112	0.264724	110.9702
113	0.261357	110.7578
114	0.257846	110.4575
115	0.254213	110.6016
116	0.243577	109.9668
117	0.242701	109.4395
118	0.242259	109.5908
119	0.230155	110.1475
120	0.22645	110.001
121	0.225397	110.0742
122	0.208896	110.4956
123	0.204266	110.4956
124	0.197299	110.2588
125	0.169315	110.127
126	0.191916	109.4946
127	0.197818	109.7925
128	0.192196	109.2578
129	0.180361	109.0723
130	0.187347	109.3726
131	0.188446	110.1074
132	0.193521	109.9146
133	0.199129	110.4321
134	0.196907	110.5713
135	0.15385	110.2051
136	0.19845	109.8706
137	0.188742	110.1636
138	0.195418	109.8877
139	0.197269	110.1587

140	0.192619	110.4468
141	0.199994	110.6494
142	0.205629	110.8276
143	0.206107	110.311
144	0.206899	110.4575
145	0.212566	110.2231
146	0.225372	110.7153
147	0.223945	110.6152
148	0.226748	110.5322
149	0.224052	110.5493
150	0.219221	110.2197
151	0.217925	109.9536
152	0.218189	109.9023
153	0.217083	109.5483
154	0.220261	109.5044
155	0.21476	110.0439
156	0.21362	110.3027
157	0.207488	110.8862
158	0.20625	110.9277
159	0.198505	110.1855
160	0.190371	109.9219
161	0.197839	109.4238
162	0.195166	109.4702
163	0.192985	110.0928
164	0.195949	109.9609
165	0.196139	110.6104
166	0.196048	111.0474
167	0.200516	110.6348
168	0.203827	110.8276
169	0.214646	110.8472
170	0.214985	111.0962
171	0.20477	110.9277
172	0.192698	110.5664
173	0.196432	110.5859
174	0.195149	110.1807
175	0.198849	109.6484
176	0.203786	109.6509
177	0.214154	109.7485
178	0.21498	110.5371
179	0.20732	109.6899
180	0.208257	109.8682
181	0.207398	110.7349
182	0.206467	110.9155
183	0.205049	110.7422
184	0.194568	110.7764
185	0.197829	110.498
186	0.192831	110.5859
187	0.195564	110.7886
188	0.193248	111.3745
189	0.19821	110.835
190	0.197386	110.9399
191	0.204166	111.25
192	0.207793	111.7529
193	0.221302	110.686
194	0.218287	110.7959
195	0.230195	111.1426
196	0.241482	110.8936
197	0.236291	111.2671
198	0.234608	110.3223
199	0.229096	110.0708
200	0.223652	109.9292
201	0.214924	110.4907
202	0.213144	110.0659
203	0.205446	109.6753
204	0.18741	109.812
205	0.1919	110.3857
206	0.19078	110.0352
207	0.181625	110.1963
208	0.19602	110.2109
209	0.195259	110.6235
210	0.182199	110.6895
211	0.194229	110.9067
212	0.186221	111.1118
213	0.197977	110.731
214	0.197673	110.6235
215	0.196605	110.3599
216	0.215061	110.3721
217	0.211342	110.1841
218	0.207965	110.4258
219	0.219648	110.1353
220	0.23121	109.9668
221	0.225315	109.8398
222	0.215815	109.2954
223	0.212665	109.5151
224	0.205414	109.73
225	0.201968	110.4956
226	0.198004	110.4858
227	0.175781	110.2319
228	0.198462	109.917
229	0.197623	109.9756
230	0.193396	109.8413
231	0.187155	109.9585
232	0.193533	109.9976
233	0.184191	109.8218
234	0.188644	109.5825
235	0.194723	109.3091
236	0.194693	109.9731
237	0.194113	110.0635
238	0.199066	109.9609
239	0.195106	110.3442
240	0.197607	110.5933
241	0.186882	110.7642
242	0.199893	111.0596
243	0.192186	110.2036
244	0.207283	110.3843
245	0.207376	110.2158
246	0.21053	110.6602
247	0.221494	110.6113
248	0.221554	110.2158
249	0.224758	110.3745
250	0.228846	110.3159
251	0.231571	110.0254
252	0.234417	110.145
253	0.231619	110.5503
254	0.233551	110.4282
255	0.230757	110.04
256	0.22711	110.1157
257	0.226972	110.1206
258	0.225526	110.6885
259	0.218113	110.6958
260	0.217039	109.9023
261	0.216822	109.6875
262	0.213572	109.8242
263	0.209513	110.1855
264	0.200101	110.3271
265	0.187347	110.4175
266	0.196197	110.4932
267	0.184448	110.498
268	0.164327	110.0366
269	0.194595	110.1611
270	0.192053	110.1709
271	0.193704	110.8984
272	0.194848	110.4639
273	0.198511	110.3735
274	0.192894	110.4834
275	0.199448	110.3027
276	0.204776	110.3076
277	0.213337	110.1807
278	0.219024	110.249
279	0.213266	109.7729
280	0.211423	109.8315
281	0.203073	109.9268
282	0.199622	109.7266
283	0.199159	109.436
284	0.197812	110.2197
285	0.195569	110.5542
286	0.190815	110.686
287	0.18208	110.0122
288	0.194424	110.4321
289	0.197835	110.8081
290	0.197353	110.5176
291	0.198401	109.8779
292	0.197406	109.6997
293	0.196149	110.2417
294	0.192465	110.4077
295	0.208522	109.8706
296	0.212493	109.9951
297	0.220533	109.7485
298	0.220682	110.9058
299	0.221905	110.3711
300	0.215041	110.481

11. Closed loop response of the liquid level process using PID controller for a pulse change of -20 mm in the level (regulatory problem).

Time (s)	FR (lit/min)	Level (mm)
0	1.204099	80.048828
1	1.204839	79.868164
2	1.208112	79.838867
3	1.202065	79.812012
4	1.207813	80.041504
5	1.20277	79.897461
6	1.2	80.058594
7	1.2047	80.107422
8	1.2	80.095215
9	1.2	80.070801
10	1.204099	79.841309
11	1.204839	79.863281
12	1.208112	79.797363
13	1.202065	79.863281
14	1.207813	79.929199
15	1.20277	79.934082
16	1.2	79.931641
17	1.2	80.007324
18	1.2	79.909668
19	1.204306	80.102539
20	1.209104	80.209961
21	1.213886	79.897461
22	1.215337	80.498047
23	1.221255	80.031738
24	1.22161	80.048828
25	1.222571	79.868164
26	1.218957	79.838867
27	1.234519	79.812012
28	1.229817	80.041504
29	1.228474	79.897461
30	1.228521	80.058594
31	1.203732	80.107422
32	1.2047	80.095215
33	1.2	80.070801
34	1.2	79.841309
35	1.204099	79.863281
36	1.204839	79.797363
37	1.208112	79.863281
38	1.202065	79.929199
39	1.207813	79.934082
40	1.20277	79.931641
41	1.2	80.007324
42	1.2	79.909668
43	1.2	80.102539
44	1.204306	79.816895
45	1.209104	79.909668
46	1.213886	79.943848
47	1.215337	80.422363
48	1.221255	80.48584
49	1.22161	80.766602
50	1.222571	80.004883
51	1.2701107	76.887207
52	1.397288	73.852539
53	1.49969	70.891113
54	1.5805335	68.193359
55	1.6804808	65.065918
56	1.780409	62.084961
57	1.79187	59.772949
58	1.789975	60.581055
59	1.822321	61.369629
60	1.834216	62.207031
61	1.83446	62.941895
62	1.8436	64.047852
63	1.849185	65.183105
64	1.959555	66.28418
65	2.006035	67.39502
66	2.031564	68.811035
67	2.059824	69.987793
68	2.075144	70.495605
69	2.081934	71.645508
70	2.147564	72.67334
71	2.227416	74.384766
72	2.228133	76.640625
73	2.230254	78.100586
74	2.235137	78.984375
75	2.247177	80.751953
76	2.407017	82.15332
77	2.479819	84.047852
78	2.489951	85.817871
79	2.542985	87.470703
80	2.564391	89.018555
81	2.584866	89.98291
82	2.58991	89.926758
83	2.588323	88.574219
84	2.575844	87.39502
85	2.572846	87.478027
86	2.561104	87.67334
87	2.545181	87.763672
88	1.8	86.82373
89	1.2	84.250488
90	1.2	81.765137
91	1.2	79.338379
92	1.2	77.053223
93	1.2	76.92627
94	1.2	76.999512
95	1.2	77.397461
96	1.2	77.76123
97	1.2	78.320312
98	1.2	78.793945
99	1.200339	79.243164
100	1.213578	79.790039
101	1.258554	79.89502
102	1.27877	80.231934
103	1.303383	80.527344
104	1.323385	80.839844
105	1.330956	81.113281
106	1.351694	81.142578
107	1.352519	81.281738
108	1.3695	81.30127
109	1.371914	81.623535
110	1.36932	81.694336
111	1.365468	81.572266
112	1.357933	81.711426
113	1.265049	82.124023
114	1.244446	82.231445
115	1.243637	82.492676
116	1.238205	82.509766
117	1.234268	82.231445
118	1.234066	82.438965
119	1.2105	82.126465
120	1.204564	82.097168
121	1.204166	82.067871
122	1.2	81.987305
123	1.2	81.89209
124	1.2	81.875
125	1.2	81.726074
126	1.2	81.696777
127	1.2	81.616211
128	1.2	81.582031
129	1.2	81.396484
130	1.2	81.245117
131	1.2	80.922852
132	1.2	81.130371
133	1.2	80.966797
134	1.2	80.800781
135	1.2	80.668945
136	1.2	80.693359
137	1.2	80.534668
138	1.2	80.595703
139	1.2	80.415039
140	1.2	80.332031
141	1.2	80.319824
142	1.2	80.327148

143	1.2	80.12207
144	1.2	79.946289
145	1.2	80.061035
146	1.2	79.931641
147	1.2	79.868164
148	1.2	79.94873
149	1.2	80.214844
150	1.2	80.280762
151	1.2	80.402832
152	1.203307	80.322266
153	1.2	80.290527
154	1.20322	80.556641
155	1.203233	80.092773
156	1.205722	79.985352
157	1.200188	79.863281
158	1.2	79.919434
159	1.2	80.112305
160	1.2	80.187988
161	1.2	80.52002
162	1.2	80.378418
163	1.2	80.593262
164	1.207352	80.50293
165	1.208531	80.432129
166	1.213627	80.422363
167	1.21303	80.251465
168	1.205376	80.336914
169	1.2	80.083008
170	1.2	80.090332
171	1.2	80.095215
172	1.2	79.675293
173	1.2	79.72168
174	1.2	79.619141
175	1.2	79.39209
176	1.2	79.606934
177	1.2	80.166016
178	1.2	80.03418
179	1.2	80.334473
180	1.203682	80.710449
181	1.204488	80.891113
182	1.208989	81.113281
183	1.216585	81.293945
184	1.223759	81.115723
185	1.217048	81.208496
186	1.212656	81.159668
187	1.206404	80.998535
188	1.205681	80.773926
189	1.200377	80.722656
190	1.2	80.634766
191	1.2	80.527344
192	1.2	80.366211
193	1.2	80.246582
194	1.2	80.180664
195	1.2	79.970703
196	1.2	80.151367
197	1.2	80.249023
198	1.2	80.244141
199	1.2	80.336914
200	1.2	80.60791
201	1.2	80.541992
202	1.2	80.571289
203	1.206856	80.441895
204	1.204466	80.317383
205	1.204402	80.253906
206	1.203462	80.129395
207	1.2	80.053711
208	1.2	80.036621
209	1.2	79.841309
210	1.2	79.870605
211	1.2	80.039062
212	1.2	80.26123
213	1.2	80.163574
214	1.2	80.15625
215	1.2	80.227051
216	1.2	80.15625
217	1.208385	80.026855
218	1.212633	79.941406
219	1.206374	79.992676
220	1.2	79.816895
221	1.2	79.780273
222	1.2	79.760742
223	1.2	80.078125
224	1.2	80.097656
225	1.2	79.96582
226	1.204772	80.622559
227	1.206052	80.795898
228	1.221216	80.822754
229	1.22167	81.035156
230	1.22703	81.228027
231	1.22163	81.325684
232	1.214379	81.369629
233	1.216422	81.237793
234	1.216153	81.311035
235	1.215398	81.118164
236	1.20605	81.052246
237	1.203233	80.981445
238	1.2	80.744629
239	1.2	80.429688
240	1.2	80.695801
241	1.2	80.639648
242	1.2	80.551758
243	1.2	80.36377
244	1.2	80.322266
245	1.2	80.15625
246	1.2	80.007324
247	1.2	80.009766
248	1.2	79.914551
249	1.2	79.833984
250	1.2	79.953613
251	1.2	80.124512
252	1.2	80.097656
253	1.2	80.170898
254	1.2	80.429688
255	1.2	80.568848
256	1.213921	80.415039
257	1.215433	80.529785
258	1.218888	80.478516
259	1.211011	80.473633
260	1.20528	80.241699
261	1.2017	80.253906
262	1.2	80.129395
263	1.2	80.065918
264	1.2	79.943848
265	1.2	79.953613
266	1.2	80.056152
267	1.2	80.214844
268	1.2	80.43457
269	1.2	80.339355
270	1.2	80.402832
271	1.2	80.588379
272	1.209214	80.537109
273	1.212387	80.393066
274	1.208574	80.551758
275	1.206256	80.258789
276	1.205835	80.466309
277	1.205678	80.163574
278	1.2	80.043945
279	1.2	80.019531
280	1.2	79.851074
281	1.2	79.765625
282	1.2	79.980469
283	1.2	80.014648
284	1.2	80.019531
285	1.2	80.249023
286	1.2	80.419922
287	1.2	80.310059
288	1.201422	80.441895
289	1.210817	80.827637
290	1.212307	81.008301
291	1.206821	81.081543
292	1.205097	81.083984
293	1.202915	81.086426
294	1.227001	80.861816
295	1.224676	80.900879
296	1.219767	80.559082
297	1.21961	80.454102
298	1.200633	80.209961
299	1.2	79.958496
300	1.2	80.178223

12. Closed loop response of the liquid level process using DMC for a step change of 30 mm in the setpoint.

Time (s)	FR (l/min)	Level (mm)
0	2	80.078027
1	2.4	80.45459
2	2.40528	81.953613
3	2.822346	83.77002
4	2.94751	86.273926
5	2.951172	89.658203
6	2.993943	90.398926
7	2.994209	93.598633
8	2.996539	94.975586
9	3	96.787109
10	3	97.912598
11	3	99.819336
12	3	100.86182
13	3	103.08838
14	3	104.46045
15	3	106.16211
16	3	107.37549
17	3	108.56445
18	2.894682	110.11231
19	2.406722	110.99219
20	2.16088	112.75635
21	2.06145	112.8501
22	0.294693	113.39111
23	0.2789	113.84033
24	0.276569	113.80371
25	0.264132	113.87695
26	0.235273	114.15772
27	0.234019	114.30176
28	0.200697	113.61328
29	0.2	114.15039
30	0.2	114.48975
31	0.2	113.83545
32	0.2	114.4458
33	0.2	113.57666
34	0.2	113.60596
35	0.2	113.77441
36	0.2	112.38281
37	0.2	112.37549
38	0.2	112.49023
39	0.2	112.39014
40	0.2	111.95801
41	0.2	111.81885
42	0.2	112.40967
43	0.2	111.46729
44	0.2	111.60156
45	0.2	110.95703
46	0.2	110.13428
47	0.2	110.46143
48	0.2	110.7959
49	0.2	110.41504
50	0.2	110.44922
51	0.2	109.91211
52	0.212806	109.76563
53	0.217536	109.4751
54	0.220339	109.37988
55	0.226011	109.03076
56	0.228549	108.88672
57	0.22965	109.28223
58	0.229914	109.83887
59	0.231146	109.04297
60	0.234324	109.84131
61	0.24349	109.00635
62	0.246704	108.51074
63	0.253598	108.98682
64	0.261908	108.76221
65	0.276429	109.11865
66	0.276498	109.74121
67	0.280975	109.40918
68	0.287208	109.43359
69	0.294571	110.2832
70	0.290519	110.26367
71	0.286659	110.59815
72	0.282707	110.4541
73	0.276936	110.34668
74	0.276376	110.06348
75	0.273142	109.93897
76	0.273481	109.92676
77	0.277894	110.42236
78	0.265822	110.92285
79	0.262255	111.67725
80	0.252553	110.79102
81	0.251402	110.77393
82	0.248843	110.96191
83	0.246794	111.5625
84	0.241859	112.16065
85	0.218879	111.48682
86	0.211954	111.604
87	0.210236	111.6626
88	0.209305	112.37305
89	0.205052	113.28369
90	0.2	112.50977
91	0.2	111.8335
92	0.2	112.88574
93	0.2	112.75147
94	0.2	111.59912
95	0.2	111.90674
96	0.2	111.87988
97	0.2	112.04834
98	0.2	111.34277
99	0.2	110.68848
100	0.2	111.08154
101	0.2	111.09375
102	0.2	111.28906
103	0.2	112.146
104	0.2	110.78369
105	0.2	109.36523
106	0.204894	110.39063
107	0.2	109.97559
108	0.202927	109.49219
109	0.205068	110.25147
110	0.2	110.5957
111	0.2	110.6543
112	0.2	110.84717
113	0.2	110.73242
114	0.2	110.92529
115	0.2	111.02051
116	0.2	110.14648
117	0.2	110.09766
118	0.2	109.56055
119	0.201205	109.62402
120	0.213503	109.73145
121	0.232768	110.0293
122	0.227186	110.25635
123	0.223467	110.05615
124	0.22009	109.88037
125	0.231773	110.40039
126	0.223652	111.20606
127	0.211863	111.12793
128	0.202362	111.05469
129	0.2	110.84717
130	0.2	110.99609
131	0.2	110.87402
132	0.2	110.97656
133	0.2	110.1294
134	0.2	110.08057
135	0.2	110.56641
136	0.2	110.07568
137	0.2	110.02686
138	0.2	110.54199
139	0.2	110.03906
140	0.2	109.99756
141	0.201215	109.80469
142	0.204295	109.87793
143	0.206903	109.73877
144	0.207395	109.59961
145	0.211113	109.62402
146	0.211156	110.14648
147	0.2	110.21729

148	0.2	109.72168	199	0.2	109.47266	250	0.2	111.23535
149	0.200156	110.56641	200	0.205139	109.57764	251	0.2	109.98779
150	0.2	110.34668	201	0.213662	109.53125	252	0.201907	110.87891
151	0.2	110.17334	202	0.217632	109.80957	253	0.2	111.25
152	0.2	110.87647	203	0.225672	110.93018	254	0.2	111.29395
153	0.2	109.74365	204	0.214101	110.61768	255	0.2	110.78369
154	0.20006	110.01221	205	0.212165	110.53711	256	0.2	110.62744
155	0.200021	109.74365	206	0.205301	110.16357	257	0.2	110.61523
156	0.204109	109.85352	207	0.20033	110.63965	258	0.2	110.62256
157	0.206835	110.07324	208	0.2	111.26709	259	0.2	110.46631
158	0.2	111.0083	209	0.2	110.46387	260	0.2	109.44824
159	0.2	110.65918	210	0.2	110.41504	261	0.202548	110.24414
160	0.2	110.72266	211	0.2	110.7544	262	0.2	109.83887
161	0.2	111.05957	212	0.2	110.34912	263	0.209708	109.646
162	0.2	110.73975	213	0.2	110.92529	264	0.209713	109.95117
163	0.2	110.49561	214	0.2	110.53477	265	0.21[illegible]48	110.05859
164	0.2	110.24902	215	0.2	110.71045	266	0.209087	110.66162
165	0.2	109.66797	216	0.2	109.93408	267	0.208446	110.30762
166	0.206027	110.07324	217	0.20388	109.85352	268	0.208327	110.48828
167	0.205919	109.97315	218	0.205379	109.69727	269	0.206827	110.43457
168	0.20815	110.05859	219	0.205644	109.57275	270	0.200449	110.39063
169	0.20612	110.38086	220	0.218291	109.92432	271	0.2	110.79834
170	0.201415	109.95606	221	0.221401	110.01465	272	0.2	111.47949
171	0.205499	110.93994	222	0.220277	109.97803	273	0.2	111.0669
172	0.203598	110.21484	223	0.224781	110.48096	274	0.2	110.85449
173	0.2	110.22705	224	0.217191	111.18408	275	0.2	110.7666
174	0.2	111.1499	225	0.2	110.46143	276	0.2	110.271
175	0.2	110.33203	226	0.2	110.60303	277	0.2	110.56152
176	0.2	110.11231	227	0.2	110.53223	278	0.2	111.18652
177	0.2	109.68262	228	0.2	111.15234	279	0.2	110.67871
178	0.202196	110.1416	229	0.2	111.00586	280	0.2	109.84375
179	0.201451	109.73877	230	0.2	110.11719	281	0.208052	109.75342
180	0.204704	109.94629	231	0.2	110.71533	282	0.211667	110.09277
181	0.206598	110.18555	232	0.2	110.15625	283	0.210244	110.65674
182	0.200314	110.30029	233	0.2	110.50049	284	0.20681	110.54688
183	0.2	110.40283	234	0.2	110.40772	285	0.206209	110.83252
184	0.2	111.14746	235	0.2	109.6167	286	0.200284	110.96924
185	0.2	110.63721	236	0.208319	109.74121	287	0.2	110.92773
186	0.2	110.76172	237	0.208792	109.4165	288	0.2	111.3794
187	0.2	109.78516	238	0.213155	109.38477	289	0.2	111.12549
188	0.201975	109.78027	239	0.219147	109.59961	290	0.2	110.91309
189	0.202736	110.37598	240	0.221068	110.52734	291	0.2	111.77002
190	0.200548	110.57129	241	0.220862	110.36621	292	0.2	111.08398
191	0.2	110.13428	242	0.21446	111.00586	293	0.2	111.12549
192	0.2	110.05371	243	0.209598	111.19141	294	0.2	111.03027
193	0.2	111.1499	244	0.207242	110.77881	295	0.2	111.03272
194	0.2	110.44678	245	0.203235	111.48193	296	0.2	110.54443
195	0.2	110.57861	246	0.2	111.05225	297	0.2	111.28174
196	0.2	109.90234	247	0.2	111.09619	298	0.2	110.95459
197	0.202414	110.25147	248	0.2	111.3501	299	0.2	110.61523
198	0.2	110.30762	249	0.2	110.21484	300	0.2	110.57861

13. Closed loop response of the liquid level process using DMC for a pulse change of – 20 mm in the level (regulatory problem).

Time (s)	FR (lit/min)	Level (mm)
0	0.2	80.447363
1	0.2	80.464453
2	0.2	80.410645
3	0.2	80.239746
4	0.2	80.134766
5	0.2	80.015137
6	0.2	80.481445
7	0.2	80.505371
8	0.2	80.463867
9	0.2	80.358887
10	0.2	80.20752
11	0.2	80.163574
12	0.2	80.251465
13	0.2	80.131836
14	0.2	79.863281
15	0.204099	80.012207
16	0.200137	79.975586
17	0.20341	80.292969
18	0.2	80.324707
19	0.2	80.227051
20	0.2	80.270996
21	0.2	80.366211
22	0.2	80.34668
23	0.2	80.310059
24	0.2	79.968262
25	0.204798	80.117188
26	0.2	80.168457
27	0.2	80.200195
28	0.2	80.241699
29	0.2	80.349121
30	0.2	80.124512
31	0.2	80.180664
32	0.2	80.014648
33	0.2	80.20752
34	0.2	80.014648
35	0.2	79.853516
36	0.206761	79.887695
37	0.213913	79.919434
38	0.21431	80.500488
39	0.210074	80.246582
40	0.204724	80.419922
41	0.203485	80.661621
42	0.2	80.847168
43	0.2	80.795898
44	0.2	80.65918
45	0.2	80.957031
46	0.2	80.827637
47	0.2	80.29248
48	0.2	79.067351
49	0.2	77.185027
50	0.2	74.885222
51	0.201407	73.032195
52	0.210895	71.037566
53	0.218389	68.955046
54	0.245613	66.726042
55	0.256893	64.929167
56	0.264908	63.293425
57	0.267472	61.596648
58	0.299852	60.876433
59	0.359737	60.36862
60	0.382237	59.999968
61	0.834826	60.517546
62	1.198408	61.884734
63	1.215072	62.480437
64	1.226753	65.612761
65	1.24868	67.915007
66	1.257378	69.936491
67	1.264397	71.633269
68	1.271099	73.686491
69	1.277389	75.76657
70	1.284887	77.636687
71	1.287039	79.821745
72	1.292709	80.168425
73	1.290783	81.647461
74	1.2891	82.453125
75	1.287071	83.317383
76	1.272591	83.917969
77	1.256734	83.847168
78	1.254848	83.805664
79	1.227358	83.622559
80	1.02	83.251465
81	0.8	83.314941
82	0.4	82.829102
83	0.2	82.765625
84	0.2	82.645996
85	0.2	82.313965
86	0.2	82.152832
87	0.2	82.414062
88	0.2	81.793945
89	0.2	81.762207
90	0.2	81.393555
91	0.2	81.120117
92	0.2	81.09082
93	0.2	80.919922
94	0.2	80.770996
95	0.2	80.802734
96	0.2	80.248535
97	0.2	80.209473
98	0.2	80.229004
99	0.2	80.167969
100	0.2	80.48584
101	0.2	80.559082
102	0.2	80.559082
103	0.213439	80.734863
104	0.207662	80.78125
105	0.201183	80.952148
106	0.200311	80.864258
107	0.2	80.59082
108	0.2	80.32959
109	0.2	80.517578
110	0.2	80.170898
111	0.2	79.848633
112	0.2	79.902344
113	0.2	79.72168
114	0.2	79.94873
115	0.205629	79.855957
116	0.206107	80.209961
117	0.206899	80.109863
118	0.212566	80.419922
119	0.225372	80.686035
120	0.223945	80.883789
121	0.221618	80.791016
122	0.218921	80.866699
123	0.21409	81.003418
124	0.212795	81.149902
125	0.211547	81.079102
126	0.21044	81.057129
127	0.2	80.847168
128	0.2	80.759277
129	0.2	80.710449
130	0.2	80.253906
131	0.2	80.603027
132	0.2	80.053711
133	0.2	80.046387
134	0.2	79.98291
135	0.2	79.938965
136	0.2	80.15625
137	0.2	80.021973
138	0.200067	79.924316
139	0.202326	80.063477
140	0.2	79.973145
141	0.2	80.083008
142	0.210819	80.324707
143	0.201949	80.065918
144	0.206362	79.946289
145	0.2	79.934082
146	0.2	79.699707
147	0.2	80.129395

148	0.2065	80.007324
149	0.210286	80.64209
150	0.220655	80.710449
151	0.218187	80.808105
152	0.210527	80.837402
153	0.207065	80.991211
154	0.206206	80.996094
155	0.205741	80.961914
156	0.204323	81.079102
157	0.2	80.932617
158	0.2	80.786133
159	0.2	80.830078
160	0.2	80.571289
161	0.2	80.612793
162	0.2	80.654297
163	0.2	80.13916
164	0.2	80.375977
165	0.2	80.34668
166	0.2	79.892578
167	0.2	79.775391
168	0.2	80.078125
169	0.2	79.987793
170	0.206493	80.214844
171	0.20726	80.495605
172	0.201749	80.217285
173	0.206642	80.214844
174	0.2	80.300293
175	0.2	79.916992
176	0.2	79.926758
177	0.2	80.13916
178	0.2	80.036621
179	0.200624	80.097656
180	0.202869	80.195312
181	0.200879	80.166016
182	0.2	79.916992
183	0.2	79.848633
184	0.2	79.956055
185	0.2	80.100098
186	0.201205	80.239258
187	0.213503	80.3125
188	0.232768	80.385742
189	0.227186	80.27832
190	0.223467	80.461426
191	0.22009	80.812988
192	0.210195	80.429688
193	0.202074	80.36377
194	0.2	80.45166
195	0.2	80.341797
196	0.2	80.285645
197	0.2	79.824219
198	0.2	80.083008
199	0.2	79.96582
200	0.2	79.829102
201	0.200591	79.973145
202	0.2	80.039062
203	0.203107	80.107422
204	0.203809	80.383301
205	0.208316	80.13916
206	0.200412	80.378418
207	0.2	79.997559
208	0.2	79.904785
209	0.2	79.724121
210	0.2	79.841309
211	0.200491	79.875488
212	0.20421	80.061035
213	0.204253	80.185547
214	0.208156	80.26123
215	0.220266	80.432129
216	0.212452	80.534668
217	0.211501	80.253906
218	0.210578	80.67627
219	0.208754	80.622559
220	0.204964	80.681152
221	0.20004	80.852051
222	0.200002	80.72998
223	0.2	80.422363
224	0.2	80.512695
225	0.2	80.195312
226	0.2	80.12207
227	0.2	80.222168
228	0.2	79.951172
229	0.2	79.84375
230	0.2	80.166016
231	0.2	79.941406
232	0.2075	79.997559
233	0.213527	79.936523
234	0.213419	80.136719
235	0.21565	80.273438
236	0.223357	80.29541
237	0.224999	80.36377
238	0.218622	80.43457
239	0.216721	80.541992
240	0.208945	80.302734
241	0.2	80.17334
242	0.2	80.34668
243	0.2	80.197754
244	0.2	79.396973
245	0.2	79.35791
246	0.2	79.816895
247	0.2	79.726562
248	0.201894	79.794922
249	0.20667	80.109863
250	0.21523	80.053711
251	0.220917	79.899902
252	0.222983	80.273438
253	0.22114	80.014648
254	0.21279	80.266113
255	0.214765	80.471191
256	0.213924	80.307617
257	0.211736	80.458984
258	0.207305	80.332031
259	0.2	79.904785
260	0.2	79.584961
261	0.2	79.780273
262	0.2	79.958496
263	0.216881	80.095215
264	0.219295	79.875488
265	0.220309	80.258789
266	0.230418	80.065918
267	0.222883	80.175781
268	0.231405	80.205078
269	0.225668	80.632324
270	0.225582	80.3125
271	0.214011	80.517578
272	0.212075	80.874023
273	0.205211	80.67627
274	0.20024	80.412598
275	0.2	80.378418
276	0.2	80.358887
277	0.2	80.339355
278	0.2	79.836426
279	0.2	80.056152
280	0.2	79.931641
281	0.2	79.816895
282	0.205339	79.787598
283	0.2	79.897461
284	0.20388	79.733887
285	0.205379	79.853516
286	0.205644	79.865723
287	0.218291	80.036621
288	0.221401	80.197754
289	0.22347	79.953613
290	0.227975	80.522461
291	0.220385	80.307617
292	0.203743	79.975586
293	0.206374	80.742188
294	0.205452	80.534668
295	0.204485	80.864258
296	0.221612	80.725098
297	0.220817	80.671387
298	0.212526	80.447363
299	0.210456	80.410645
300	0.200391	80.239746

14. Closed loop response of the liquid level process using WDMC for a step change of 30 mm in the setpoint.

Time (s)	FR (l/min)	Level (mm)
0	2.510421	80.133301
1	2.850421	81.335938
2	2.998535	82.480957
3	3	85.007812
4	3	86.94873
5	3	88.750488
6	3	91.069824
7	3	93.547852
8	2.994543	95.388672
9	2.965627	97.698242
10	2.719785	98.623535
11	2.620356	99.858887
12	2.267013	101.44092
13	1.986715	103.4502
14	1.555947	104.99316
15	1.153072	106.18457
16	0.930378	107.99609
17	0.881213	108.0791
18	0.682751	107.99121
19	0.550957	108.50879
20	0.305403	108.52344
21	0.31457	109.13135
22	0.317783	109.65381
23	0.31165	109.51953
24	0.310413	110.05176
25	0.30654	110.35693
26	0.301726	110.26904
27	0.300645	110.48633
28	0.295811	110.46436
29	0.288797	110.66943
30	0.280694	110.62061
31	0.272973	111.07471
32	0.265069	111.32617
33	0.253526	111.21631
34	0.252406	110.98682
35	0.245939	110.87451
36	0.228199	111.5044
37	0.2	111.7583
38	0.2	111.9292
39	0.2	111.79004
40	0.2	111.60205
41	0.2	111.43115
42	0.2	111.30664
43	0.2	111.55566
44	0.2	111.6167
45	0.2	111.41895
46	0.2	110.9917
47	0.2	111.69727
48	0.2	111.75342
49	0.2	111.90234
50	0.2	111.69727
51	0.2	111.6167
52	0.2	111.39453
53	0.2	111.28711
54	0.2	111.00879
55	0.2	111.13818
56	0.2	111.07715
57	0.2	111.27002
58	0.2	111.36523
59	0.2	111.24072
60	0.2	111.03076
61	0.2	111.36768
62	0.2	111.24805
63	0.2	111.26514
64	0.2	110.88672
65	0.2	110.63525
66	0.2	111.19434
67	0.2	111.26758
68	0.2	111.07227
69	0.2	111.1333
70	0.2	111.05029
71	0.2	110.95752
72	0.2	110.7915
73	0.2	110.52051
74	0.2	110.7085
75	0.2	110.9502
76	0.2	110.94531
77	0.2	111.04297
78	0.2	110.71094
79	0.2	110.91602
80	0.2	110.73535
81	0.2	110.17139
82	0.2	110.65723
83	0.2	110.87695
84	0.2	110.35449
85	0.2	110.86719
86	0.2	111.04541
87	0.2	110.78174
88	0.2	110.54248
89	0.2	110.34229
90	0.2	110.57666
91	0.2	110.40088
92	0.2	110.43506
93	0.2	110.37158
94	0.2	110.4375
95	0.2	109.85645
96	0.2	110.03955
97	0.2	110.22754
98	0.2	109.67822
99	0.2	110.06397
100	0.2	109.59766
101	0.2	109.67334
102	0.2	109.6294
103	0.2	109.85156
104	0.2	110.0957
105	0.2	110.1665
106	0.2	109.77588
107	0.2	109.59766
108	0.2	109.39746
109	0.200156	109.71729
110	0.2	109.21191
111	0.200093	109.59033
112	0.2	109.70752
113	0.2	109.64648
114	0.2	109.771
115	0.2	109.52441
116	0.2	109.77832
117	0.2	109.5
118	0.202846	109.30713
119	0.203381	109.64404
120	0.2	109.74902
121	0.2	109.80029
122	0.2	109.51709
123	0.2	109.48291
124	0.201741	109.28027
125	0.209241	109.41211
126	0.215268	109.61963
127	0.21516	109.63184
128	0.213535	109.99072
129	0.211505	110.00537
130	0.2068	110.16162
131	0.200423	110.38867
132	0.2	110.6084
133	0.2	110.68164
134	0.2	110.55225
135	0.2	110.63525
136	0.2	110.01758
137	0.2	109.9419
138	0.2	110.22266
139	0.2	110.10547
140	0.2	109.98828
141	0.2	109.84912
142	0.2	109.93213
143	0.2	109.89795
144	0.2	110.30811
145	0.2	110.33984
146	0.2	110.24707
147	0.2	110.14697
148	0.2	110.24463
149	0.2	109.98828
150	0.2	109.95898

151	0.2	110.43506	201	0.2	110.24951	251	0.2	110.18115
152	0.2	110.2959	202	0.2	110.2544	252	0.2	110.08106
153	0.2	110.38867	203	0.2	110.32031	253	0.2	110.2544
154	0.2	110.22266	204	0.2	110.22754	254	0.2	110.17383
155	0.2	109.89307	205	0.2	109.89551	255	0.2	110.07617
156	0.2	110.44971	206	0.2	109.78076	256	0.2	110.26172
157	0.2	109.82471	207	0.2	109.81006	257	0.2	110.28857
158	0.2	110.21777	208	0.2	109.55371	258	0.2	110.3252
159	0.2	110.61328	209	0.2	110.22754	259	0.2	110.23975
160	0.2	110.41065	210	0.2	110.19824	260	0.2	110.41797
161	0.2	110.33496	211	0.2	110.08106	261	0.2	110.4668
162	0.2	110.21289	212	0.2	109.19482	262	0.2	110.00049
163	0.2	110.41553	213	0.205446	109.99561	263	0.2	109.854
164	0.2	110.35693	214	0.201822	110.02002	264	0.2	109.86621
165	0.2	110.43262	215	0.2	110.05176	265	0.2	110.26904
166	0.2	110.29346	216	0.2	109.79541	266	0.2	109.7417
167	0.2	110.16162	217	0.2	109.77832	267	0.2	109.86133
168	0.2	110.14453	218	0.2	109.771	268	0.2	109.76367
169	0.2	110.26172	219	0.2	110.20801	269	0.2	109.86133
170	0.2	109.96387	220	0.2	110.06641	270	0.2	109.60498
171	0.2	109.52197	221	0.2	109.87354	271	0.2	109.59522
172	0.2	110.19824	222	0.2	110.21777	272	0.2	109.6709
173	0.2	110.47412	223	0.2	110.43994	273	0.2	109.76856
174	0.2	110.10059	224	0.2	110.16162	274	0.2	110.2251
175	0.2	110.396	225	0.2	109.78076	275	0.2	110.03467
176	0.2	110.24951	226	0.2	109.93945	276	0.2	109.72461
177	0.2	110.11279	227	0.2	110.24463	277	0.2	109.67822
178	0.2	110.0542	228	0.2	110.04443	278	0.2	109.63428
179	0.2	110.32031	229	0.2	109.99072	279	0.2	109.60742
180	0.2	110.00293	230	0.2	110.1665	280	0.2	109.76123
181	0.2	110.31055	231	0.2	109.65625	281	0.2	109.83447
182	0.2	109.93945	232	0.2	109.67822	282	0.2	110.28857
183	0.2	110.02979	233	0.2	109.99072	283	0.2	110.10059
184	0.2	110.44238	234	0.2	109.50244	284	0.2	109.71973
185	0.2	110.35449	235	0.2	109.34863	285	0.2	109.77344
186	0.2	110.07129	236	0.200803	109.68799	286	0.2	110.28369
187	0.2	109.69287	237	0.200702	109.46582	287	0.2	110.01758
188	0.2	110.19092	238	0.20886	109.98096	288	0.2	109.84912
189	0.2	110.23975	239	0.205892	110.30322	289	0.2	109.56104
190	0.2	110.47412	240	0.205759	110.09815	290	0.2	109.72461
191	0.2	110.13477	241	0.200372	110.49121	291	0.2	110.06641
192	0.2	109.97607	242	0.2	110.49854	292	0.2	110.00293
193	0.2	109.67822	243	0.2	110.47656	293	0.2	110.32031
194	0.2	109.88086	244	0.2	110.27148	294	0.2	109.65381
195	0.2	110.14453	245	0.2	110.22022	295	0.2	109.33398
196	0.2	109.96143	246	0.2	109.96143	296	0.200713	109.57813
197	0.2	110.30811	247	0.2	110.03467	297	0.2	109.65625
198	0.2	109.9834	248	0.2	110.41309	298	0.2	109.52197
199	0.2	109.9419	249	0.2	110.15186	299	0.2	109.95166
200	0.2	110.11035	250	0.2	110.1665	300	0.2	109.87354

15. Closed loop response of the liquid level process using WDMC for a pulse change of –20 mm in the level (regulatory problem).

Time (s)	FR (l/min)	Level (mm)
0	0.2	80.449219
1	0.2	80.310059
2	0.2	80.517578
3	0.2	80.390625
4	0.2	80.371094
5	0.2	80.314941
6	0.2	80.266113
7	0.2	80.158691
8	0.2	80.378418
9	0.2	80.229492
10	0.2	80.283203
11	0.2	80.202637
12	0.2	80.15625
13	0.2	79.980469
14	0.207707	80.214844
15	0.203001	80.439453
16	0.2	80.529785
17	0.2	80.79834
18	0.2	80.834961
19	0.2	80.070801
20	0.2	80.546875
21	0.2	80.625
22	0.2	80.632324
23	0.2	80.749512
24	0.2	80.673828
25	0.2	80.683594
26	0.2	80.67627
27	0.2	80.734863
28	0.2	80.664062
29	0.2	80.600586
30	0.2	80.473633
31	0.2	80.546875
32	0.2	80.776367
33	0.2	80.710449
34	0.2	80.688477
35	0.2	80.605469
36	0.2	80.539551
37	0.2	80.705566
38	0.2	80.585938
39	0.2	80.64209
40	0.2	80.74707
41	0.2	80.588379
42	0.2	80.769043
43	0.2	80.588379
44	0.2	80.732422
45	0.2	80.76416
46	0.2	80.571289
47	0.2	80.76498
48	0.2	78.326016
49	0.201563	75.828457
50	0.209601	73.091641
51	0.228613	70.420742
52	0.256331	68.21127
53	0.438145	65.869961
54	0.617716	63.474941
55	2.463674	61.778164
56	2.657344	60.427114
57	2.936406	60.00082
58	2.940976	60.572109
59	2.943768	61.064211
60	2.950827	63.221035
61	2.948006	65.79916
62	2.932821	67.801113
63	2.90743	71.26791
64	2.901547	73.738613
65	2.791605	76.165371
66	1.52	79.002285
67	0.7	81.097012
68	0.2	83.826504
69	0.2	83.811855
70	0.2	82.910977
71	0.2	82.163906
72	0.2	81.369629
73	0.2	81.289062
74	0.2	81.333008
75	0.2	80.142578
76	0.2	80.296387
77	0.2	80.213379
78	0.2	80.347656
79	0.2	80.291504
80	0.2	80.09375
81	0.2	80.171875
82	0.2	80.181641
83	0.2	80.196289
84	0.2	80.040039
85	0.2	80.374512
86	0.2	80.220703
87	0.2	80.271973
88	0.2	80.208496
89	0.2	80.005859
90	0.2	80.188965
91	0.2	80.242676
92	0.2	80.25
93	0.2	80.062012
94	0.2	80.208496
95	0.2	80.286621
96	0.2	80.257324
97	0.2	80.276855
98	0.2	80.28418
99	0.2	80.188965
100	0.2	80.237793
101	0.2	80.325684
102	0.2	80.411133
103	0.2	80.179199
104	0.2	80.088867
105	0.2	80.11084
106	0.2	80.027832
107	0.2	80.07666
108	0.2	80.19873
109	0.2	80.201172
110	0.2	80.296387
111	0.2	80.071777
112	0.2	80.057129
113	0.2	80.147461
114	0.2	80.064453
115	0.2	80.079102
116	0.2	80.022949
117	0.2	80.091309
118	0.2	80.264648
119	0.2	80.120605
120	0.2	80.140137
121	0.2	79.993652
122	0.2	80.064453
123	0.2	80.157227
124	0.2	80.018066
125	0.2	80.108398
126	0.2	80.159668
127	0.2	80.274414
128	0.2	79.935059
129	0.2	80.003418
130	0.2	80.147461
131	0.2	80.105957
132	0.2	79.876465
133	0.2	80.030273
134	0.2	80.101074
135	0.2	79.949707
136	0.2	79.769043
137	0.2	80.057129
138	0.2	79.854492
139	0.2	80.203613
140	0.2	80.003418
141	0.2	79.90332
142	0.2	79.876465
143	0.2	79.791016
144	0.2	80.003418
145	0.2	80.081543
146	0.2	79.917773
147	0.2	80.052051

148	0.2	80.030078	199	0.2	79.903125	250	0.2	80.017871
149	0.2	80.142383	200	0.2	79.895801	251	0.2	79.846973
150	0.2	80.098438	201	0.2	79.739551	252	0.2	79.805469
151	0.2	80.066699	202	0.2	79.79082	253	0.2	79.873828
152	0.2	80.149707	203	0.2	79.839648	254	0.2	79.900684
153	0.2	79.983691	204	0.2	79.781055	255	0.2	79.715137
154	0.2	80.010547	205	0.2	79.817676	256	0.2	79.820117
155	0.2	80.144824	206	0.2	79.622363	257	0.2	79.695605
156	0.2	80.08623	207	0.2	79.900684	258	0.2	79.851855
157	0.2	79.942188	208	0.2	79.903125	259	0.2	79.658984
158	0.2	80.113086	209	0.2	79.856738	260	0.2	79.671191
159	0.2	79.920215	210	0.2	79.839648	261	0.2	79.734668
160	0.2	80.003223	211	0.2	79.966602	262	0.2	79.798145
161	0.2	79.868945	212	0.2	80.056934	263	0.2	79.695605
162	0.2	80.017871	213	0.2	79.85918	264	0.2	79.688281
163	0.2	80.179004	214	0.2	79.820117	265	0.2	79.754199
164	0.2	79.851855	215	0.2	79.754199	266	0.2	79.741992
165	0.2	80.205859	216	0.2	79.881152	267	0.2	79.65166
166	0.2	80.027637	217	0.2	79.991016	268	0.2	79.666309
167	0.2	79.822559	218	0.2	79.771289	269	0.2	79.746875
168	0.2	79.954395	219	0.2	79.873828	270	0.2	79.925098
169	0.2	79.966602	220	0.2	79.820117	271	0.2	79.595508
170	0.2	80.027637	221	0.2	80.064258	272	0.2	79.820117
171	0.2	79.827441	222	0.2	79.77373	273	0.2	79.712695
172	0.2	79.922656	223	0.2	79.744434	274	0.2	80.234375
173	0.2	79.817676	224	0.2	79.763965	275	0.2	80.351562
174	0.2	79.912891	225	0.2	79.920215	276	0.2	80.371094
175	0.2	79.77373	226	0.2	79.829883	277	0.2	80.375977
176	0.2	80.147266	227	0.2	79.781055	278	0.2	80.373535
177	0.2	79.976367	228	0.2	79.920215	279	0.2	80.319824
178	0.2	79.954395	229	0.2	79.798145	280	0.2	80.214844
179	0.2	79.783496	230	0.2	79.800586	281	0.2	80.070801
180	0.2	79.908008	231	0.2	79.800586	282	0.2	80.129395
181	0.2	79.908008	232	0.2	79.861621	283	0.2	80.009766
182	0.2	79.934863	233	0.2	79.864062	284	0.2	80.336914
183	0.2	79.759082	234	0.2	79.881152	285	0.2	80.095215
184	0.2	79.803027	235	0.2	80.095996	286	0.2	80.209961
185	0.2	79.890918	236	0.2	79.856738	287	0.2	80.15625
186	0.2	79.851855	237	0.2	79.881152	288	0.202802	80.031738
187	0.2	80.042285	238	0.2	79.785938	289	0.2	79.995117
188	0.2	79.654102	239	0.2	79.956836	290	0.2	80.01709
189	0.2	79.72002	240	0.2	79.949512	291	0.2	80.114746
190	0.2	79.912891	241	0.2	79.995898	292	0.2	80.148926
191	0.2	79.94707	242	0.2	79.832324	293	0.2	80.200195
192	0.2	79.412402	243	0.2	79.795703	294	0.2	80.158691
193	0.2	79.568652	244	0.2	79.583301	295	0.2	80.283203
194	0.2	80.08623	245	0.2	79.793262	296	0.2	80.334473
195	0.2	80.142383	246	0.2	79.729785	297	0.2	80.141602
196	0.2	79.922656	247	0.2	79.966602	298	0.203388	80.078125
197	0.2	79.932422	248	0.2	79.949512	299	0.205826	79.98291
198	0.2	79.519824	249	0.2	79.851855	300	0.2	79.995117

BIOGRAPHIES

Biography of the Candidate

Hare Krishna Mohanta has completed his B.E. degree in Chemical Engineering from R. E. C. Rourkela (Now N.I.T. Rourkela) in the year 1995. He obtained his M.Tech degree in Chemical Engineering from I.I.T. Kanpur in the year 1998. He has 7 years of teaching experience and one year of industrial experience. He worked in Indian Rare Earths Ltd., a division of Department of Atomic Energy, Govt. of India. Currently he is working in BITS Pilani as Lecturer. He has guided two M.E. students for their Dissertations. He has designed and taught the course "Process Plant Safety" for the undergraduate Chemical Engineering students in BITS Pilani. Other courses he has taught include "Petroleum Refining and Petrochemicals", "Chemical Process Calculations", "Thermodynamics", "Environmental Pollution Control", "Systems Modeling and Analysis", "Raw Materials and Process Selection". His research interests are "Applied Wavelet Analysis", "Advanced Process Control", "Process Identification" and "Consciousness Studies". He is a Life Associate Member of Indian Institute of Chemical Engineers (IIChE).

Biography of the Supervisor

Professor Ram Krishna Gupta has completed his B.Sc. Engineering degree in Chemical Engineering in the year 1970 from H.B.T.I. Kanpur and subsequently completed M.Tech degree in the same discipline in 1973 from I.I.T. Kanpur. He also obtained Ph.D. degree from I.I.T. Kanpur in 1979. He has 25 years of teaching and 5 years of Industrial experience. He served 5 years in I.I.T. Kanpur as a research and senior research assistant. He worked as Lecturer in the University of Roorkee for 3 years and as Research Engineer in Engineers India Limited (EIL) for 5 years. He worked as Professor in University of Basrah, Iraq for 5 years. He served BITS Pilani as an Associate Professor and later as Professor for 12 years. He was the Group Leader in Chemical Engineering Group. Currently he is working as the Head of the Institute at Icfai Institute of Science and Technology, Bangalore since last 2 years. He has published several research papers in international and national journals. While in BITS

Pilani he has organized seminars on topics "*Process Control and Simulation for Process Industries*", "*Environmental Pollution and its Control*", "*Novel Separation Techniques*". He also organized intensive courses for industrial professionals on these topics. He has completed 6 research projects and supervised one Ph.D. thesis. He acted as experts in UGC 10th Plan Assessment Committee, Rajasthan Public Service Commission Selection Committee, Thapar Institute of Engineering and Technology Patiala Selection Committee, B. S. Engineering College Agra Selection committee and in Syllabus Upgrading Committee for Technical Colleges of Punjab. He also acted as External Examiner for several Universities and was member of different committee such as Research Board, Doctorate Counseling, Senate, Library, Purchase, Recruitment, etc at BITS, Pilani. Recently he is involved in setting up Icfai Institute of Science and Technology.

LIST OF PUBLICATIONS

Mohanta, H.K. and R.K Gupta, "Application of Wavelet Transform in Controls: A review", The ICFAI Journal of Science & Technology, Vol. 1, No.1, pp. 7-26, March 2005.

Mohanta, H.K. and R.K. Gupta, "Wavelet-Based Blocking and Condensing Design Methodology in Dynamic Matrix Control of a Heat Exchanger", In Proceedings of the International Conference on Emerging Technologies in Intelligent System and Control (EISCO – 2005) held in Coimbatore during 5-7 January 2005, Vol-1, pp. 283-288, Ed. V. Gunaraj, Allied Publishers, New Delhi (2005).

Mohanta, H.K. and R.K. Gupta, "Online Tuning of a Wavelet-based Dynamic Matrix Controller", In Proceedings of the 58th Annual Session of the Indian Institute of Chemical Engineers (CHEMCON – 2005) held in New Delhi during 14-17 December 2005.

Mohanta, H.K., P. N. Sheth and R.K.Gupta "Wavelet-based Model Predictive Control of a Heat Exchanger Unit", Computers and Chemical Engineering, (Communicated).

ERRATA

Submitted by: Basudeb Munshi

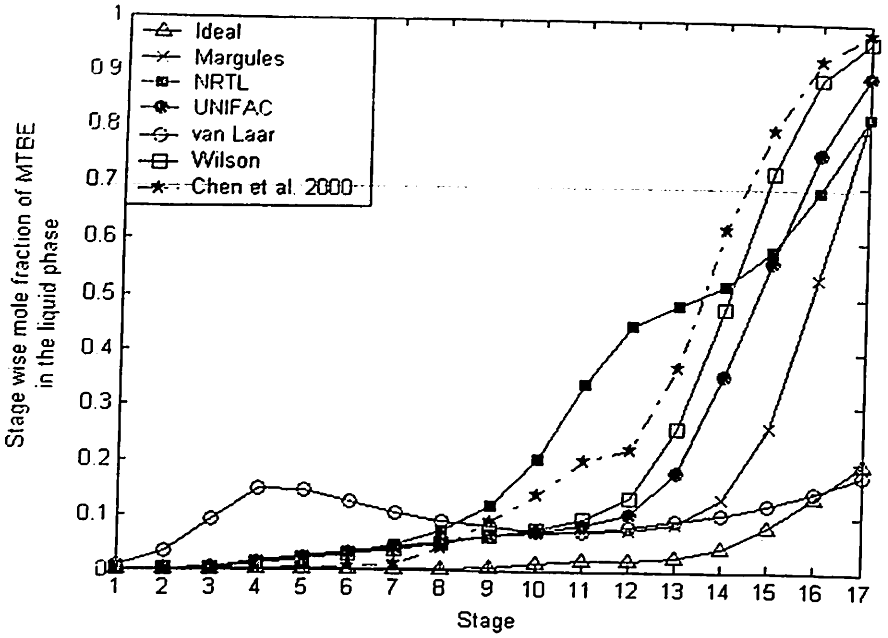


Figure 4.1 Effect of different activity models on the stage wise MTBE concentration. Vapour phase is ideal and heat effect model is used at conditions of Table 4.1. r = 6 and s = 15.

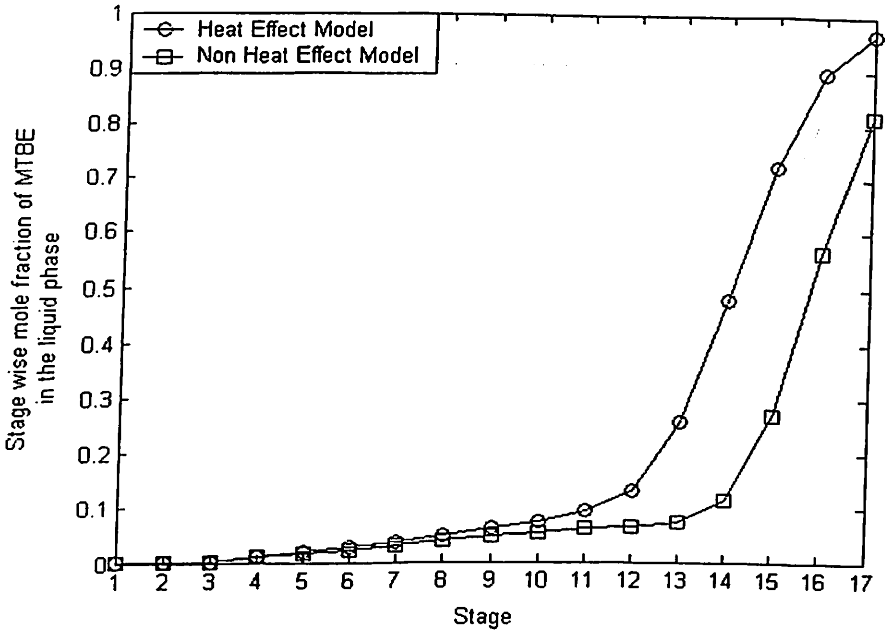


Figure 4.2 Comparison of composition profiles of MTBE between heat effect and non-heat effect model at conditions of Table 4.1. Wilson model is used. r = 6 and s = 15.

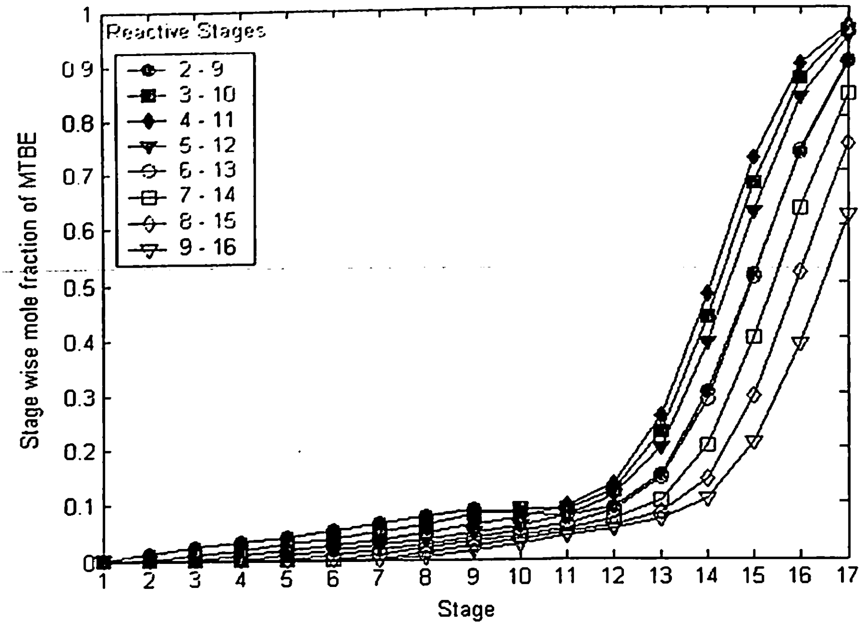


Figure 4.5 Effect of reactive stages position on the MTBE purity in the bottoms. Methanol and mixed butenes are fed at stage 10 and 11 respectively. N = 17, r = 6, s = 15. Wilson model is used.

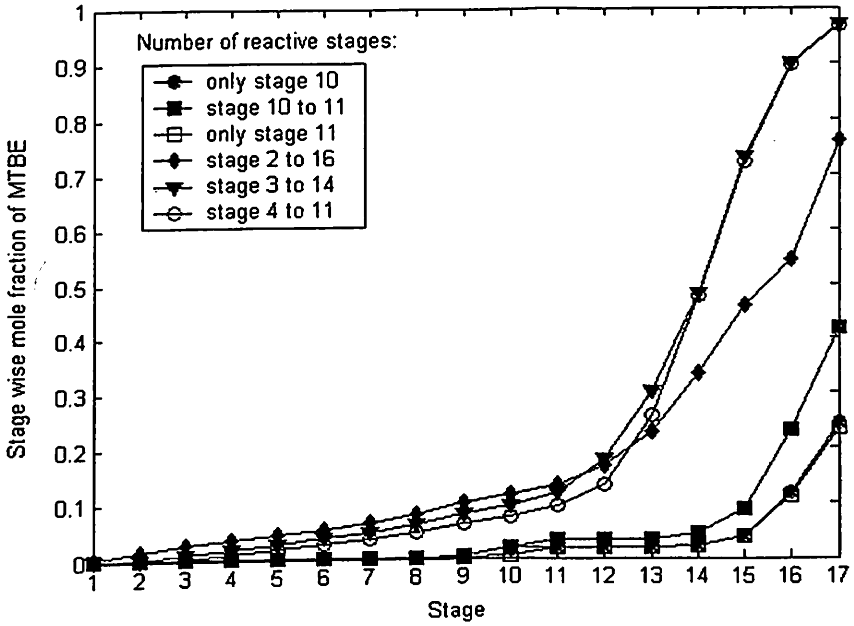


Figure 4.7 Effect of the number of reactive stage on the MTBE purity in the bottoms. Methanol and mixed butenes are fed at stage 10 and 11 respectively. N = 17, r = 6, s = 15. Wilson model is used.

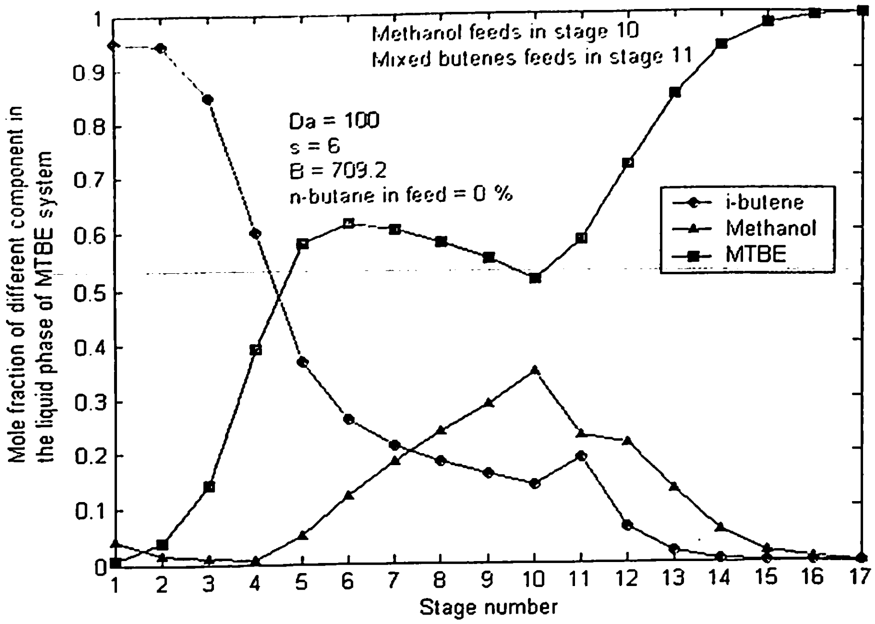


Figure 5.10 Stage wise liquid phase composition profiles in absence of n-butane in the feed at. Stages 1-3 and 12-17 are non-reactive; 4-11 stages are reactive. Simulation is done with the heat effect model and using Wilson activity model.

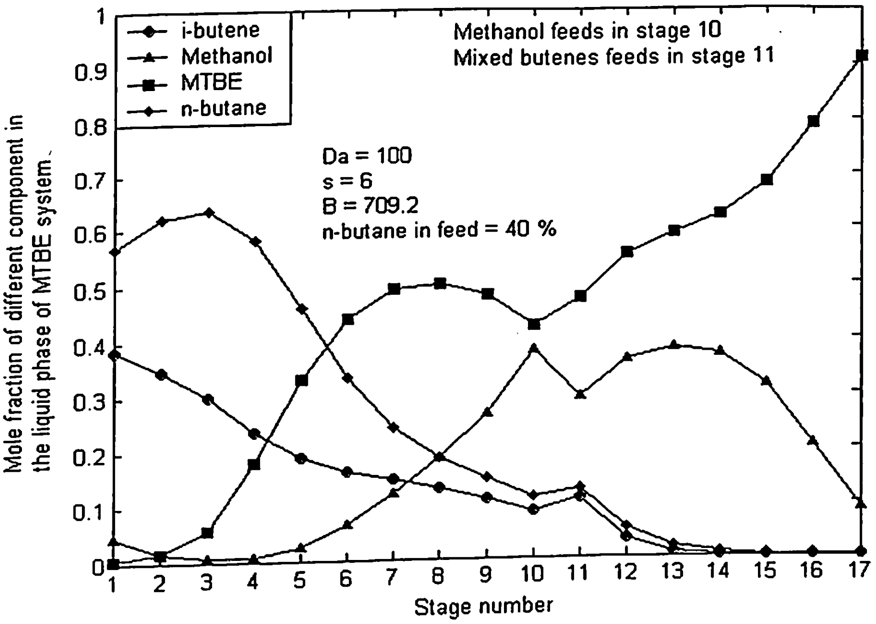


Figure 5.11 Stage wise liquid phase composition profiles in presence of n-butane of 40-mole % in the mixed butenes feed. Stages 1-3 and 12-17 are non-reactive; 4-11 stages are reactive. Simulation is done with the heat effect model and using Wilson activity model.

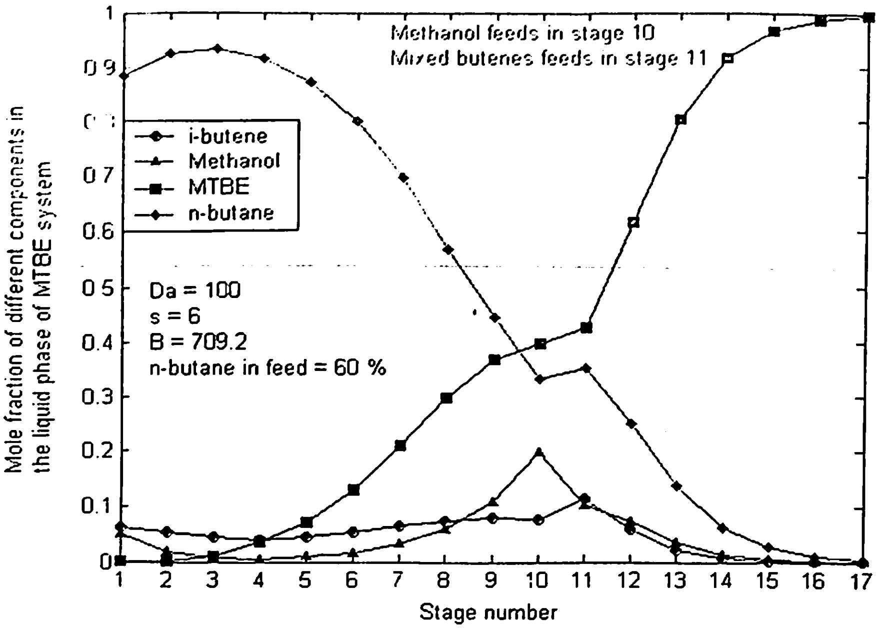


Figure 5.12 Stage wise liquid phase composition profiles in presence of n-butane of 60-mole % in the mixed butenes feed at. Stages 1-3 and 12-17 are non-reactive; 4-11 stages are reactive. Simulation is done with the heat effect model and using Wilson activity model.

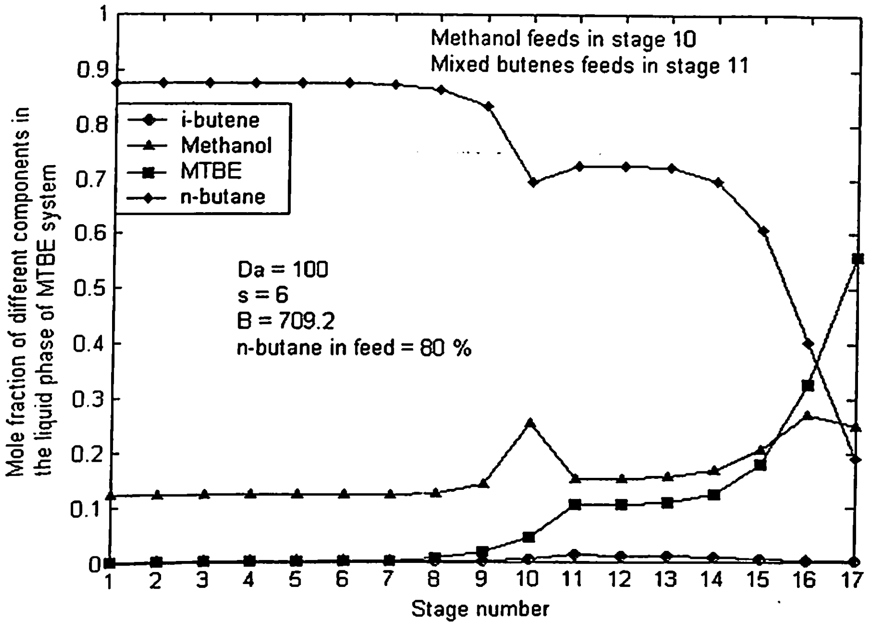


Figure 5.13 Stage wise liquid phase composition profiles in presence of n-butane of 80-mole % in the mixed butenes feed at. Stages 1-3 and 12-17 are non-reactive; 4-11 stages are reactive. Simulation is done with the heat effect model and using Wilson activity model.

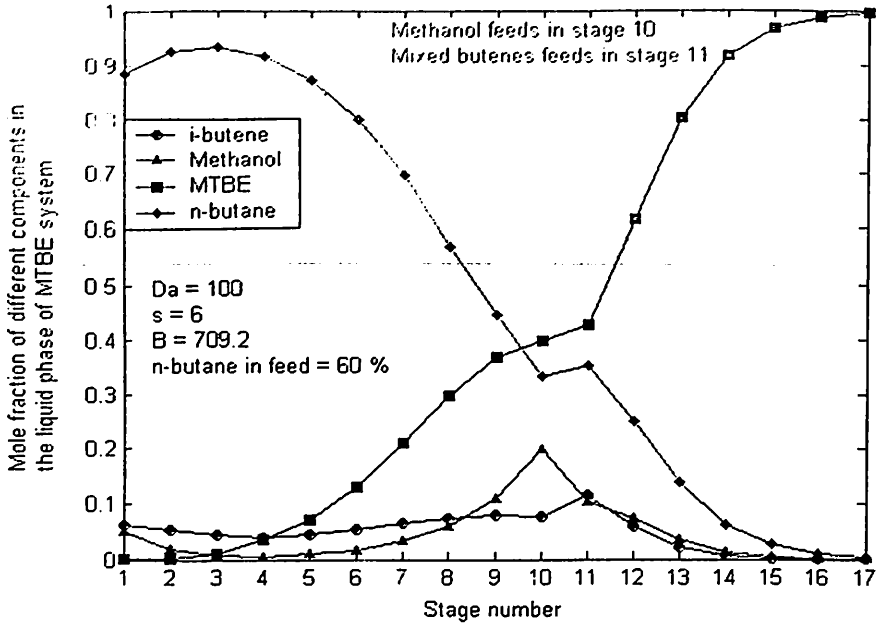


Figure 5.12 Stage wise liquid phase composition profiles in presence of n-butane of 60-mole % in the mixed butenes feed at. Stages 1-3 and 12-17 are non-reactive; 4-11 stages are reactive. Simulation is done with the heat effect model and using Wilson activity model.

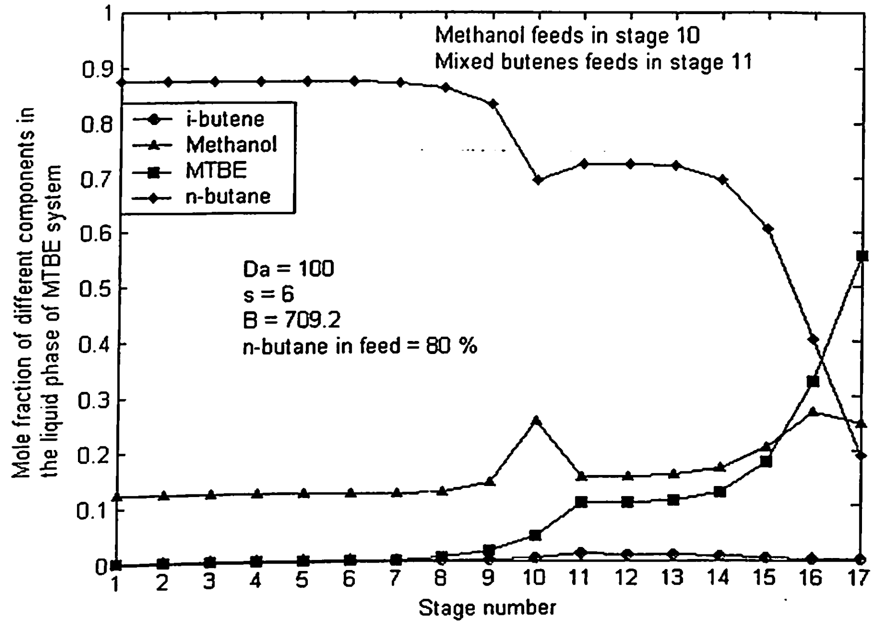


Figure 5.13 Stage wise liquid phase composition profiles in presence of n-butane of 80-mole % in the mixed butenes feed at. Stages 1-3 and 12-17 are non-reactive; 4-11 stages are reactive. Simulation is done with the heat effect model and using Wilson activity model.