55731

**ISTANBUL TECHNICAL UNIVERSITY★ INSTITUTE FOR NUCLEAR ENERGY**

55731

# NEURAL NETWORKS AND FUZZY SYSTEMS FOR ADVANCED CONTROLLER DESIGN IN NUCLEAR POWER PLANTS

M.S. THESIS

F. EROL SAĞIROĞLU. B.S. E.E.

Date of Submission : March 11. 1996

Date of Approval : March 26, 1996

Examining Committee

|  |  |  |
|---|---|---|
| Supervisor | : Prof.Dr.Melih Geçkinli | [illegible] 26.03.1996 |
| Member | : Prof.Dr.Şarman Gençay | [illegible] 26.03.1996 |
| Member | : Prof.Dr.Atilla Özgener | [illegible] 26.03.1996 |

MARCH 1996

# PREFACE

Fuzzy systems and neural networks are being studied extensively as emerging technologies in artificial intelligence(AI) applications to increase a given system's IQ. On the other hand, nuclear power plants are known as plants modeled with stiff, non-linear equations. Also including time-delay properties , nuclear power plants appear as a challenging field for control engineering. In spite of these difficulties, conventional control systems have been employed in nuclear power plants. All the required and desired operation has been carried out by these controllers with a satisfactory performance. But desired performance is limited in a simple hardware solution by the hard-wired equipment; essentially due to highly desired reliable operation of nuclear reactors. However some recent developments in digital technology (specially in microprocessors and their implementations in control ) made more sophisticated controllers applicable for a reliable nuclear reactor operation. During the years conventional controllers are in service, we have acquired a huge numerical and linguistic information stack about the plants. We need additional tools to combine this acquired information in controller design. Fuzzy logic controller has been developed to formulate computational control rules using linguistic knowledge about the plants. Neural networks can learn very complex control surfaces easily. Hence, they have been successively employed in the intelligent controller design with an even increasing application rate. Consequently, fuzzy and neural methodologies can be utilized in the design of super controllers for nuclear power plants.

I wish to express my sincere thanks to my advisor Prof. Dr. Melih Geçkinli for suggesting me a highly interesting and motivating subject, his outstanding suggestions and thoughts were always useful and influential on this work and my life. I wish also to express my sincere thanks to Prof. Dr. Şarman Gençay for his encouragement and support throughout my work on my thesis.

## Contents

Preface....................................................................................................i

Contents....................................................................................................ii

Summary....................................................................................................vi

Özet....................................................................................................vii

List of Figures.................................................................................................... xii

Chapter One: Overview.................................................................................... 1

1.1 Motivation.................................................................................... 1

1.2 Objective.................................................................................... 3

1.3 Control Methodologies.................................................................................... 3

1.4 Organization.................................................................................... 4

Chapter Two: Dynamics of Nuclear Power Plants and Classical

Control Methods.................................................................................... 8

2.1 Nomenclature.................................................................................... 8

2.2 Simulation Model.................................................................................... 9

2.3 Linearization of Nonlinear Systems....................................................................11

2.4 Linearized Model of Nuclear Plants....................................................................13

2.4.1 System Parameters Dependency on $n_{r0}$....................................................14

2.5 Classical Output Control (COC)....................................................................14

2.5.1 Inherent Nonlinearity of Plants....................................................................17

2.5.2 Results for COC.................................................................................... 21

2.5.3 Steady State Error in COC.................................................................... 26

2.5.4 Stiffness in Nuclear Power Plant Model.................................................... 26

2.5.5 Conclusion....................................................................................27

2.6 State Feedback Control with State Estimation (SFCSE)................................ 27

2.6.1 State Feedback Control with Pole Placement.................................... 27

2.6.2 State Estimation.................................................................................... 30

2.6.3 The Design of State Feedback with State Estimation..............................31

2.6.4 Simulation of SFCSE for Nuclear Power Plant.................................... 34

2.6.5 Conclusion.................................................................................... 39

Chapter Three: Optimal Control.................................................................................... 40

3.1 Linear Regulator Problem.................................................................................... 40

3.2 Hamilton-Jacobi Equation .......... 42

3.3 Solution of Optimal Control Problem .......... 44

3.1.1 Discrete Time Systems .......... 44

3.1.1.1 Discrete Time-Continuos Time Conversions .......... 47

3.3.2 Continuos Time Systems .......... 48

3.3.3 Time-Invariant Plants .......... 48

3.4 Stability of Dynamical Systems .......... 49

3.4.1 Lyapunov Stability Theory .......... 50

3.5 Simulations results of Optimal Control .......... 51

3.6 Stiffness Arising in Optimal Control for Nuclear Power Plant .......... 56

3.7 Conclusion .......... 58

Chapter Four: Stochastic Optimal Control: Robust Controller Design .......... 59

4.1 Stochastic Processes .......... 59

4.2 Discrete-Time Kalman Filter .......... 64

4.3 Classical Output Control with Kalman Filter .......... 67

4.4 Stochastic Optimal Estimator Design .......... 70

4.4.1 Time-Invariant Case .......... 72

4.5. Linear Quadratic Qaussian (LQG) Stochastic Control Problem .......... 73

4.6 Simulation Results and Conclusion .......... 74

Chapter Five: Automatically Tuned Fuzzy Logic Controller for Nuclear Power Plants .......... 80

5.1 Fuzzy Logic versus Bayesian Approach .......... 80

5.2 Fuzzy Set .......... 81

5.3 Fuzzy Logic Systems .......... 84

5.4 Fuzzy Logic Control .......... 85

5.4.1 Control with Linguistic Variables .......... 85

5.5 Classification of Fuzzy Logic Controllers .......... 86

5.6 Fuzzy Logic Controller for Nuclear Power Plant .......... 87

5.6.1 Kalman Filter Approach to Identify The Parameters of Fuzzy Rules .......... 89

5.6.2 Least Square Estimation of Fuzzy Rules .......... 90

5.6.3 Tuning Mechanism .......... 91

5.6.4 Identification of Fuzzy Rules.....92
5.7 Simulation of FLC for Nuclear Power Plant..... 94
5.8 Conclusion..... 97
Chapter Six: Artificial Neural Networks
and Neurocontrol of Nuclear Power Plants..... 99
6.1 Artificial Neural Networks..... 99
6.2 Neural Networks and Learning Algorithms..... 100
6.3 Neural Network Structures ..... 102
6.4 Adaptation of Neural Networks..... 103
6.5 Parameter Level Adaptation..... 104
6.5.1 Backpropagation (BP) Algorithm for FNN..... 104
6.5.2 Least Mean Square(LMS) Algorithm..... 105
6.5.3 Learning Algorithm vía Extended Kalman Filter..... 105
6.5.4 Other Algorithms and Improving Learning.....106
6.7 Important Theorems in Neural Networks.....106
6.7.1 Kalmogorov's Theorem and
Its Application to Neural Networks..... 107
6.7.2 The Stone-Weierstrass Theorem..... 107
6.8 Neural Networks in System Identification and Control..... 107
6.9 Neurocontrol of Nuclear Power Plants..... 111
6.10 Conclusion..... 119
Chapter Seven: Conclusions and Discussions.....120
7.1 Summary of Work..... 120
7.2 Comparison of Controllers..... 121
7.2.1 Structural Comparison.....121
7.2.2 Intelligent Control..... 122
7.2.3 Robustness..... 122
7.3 Conclusions and Future Work..... 122

References........................................................................................................124

Appendix A:Simulation of Nonlinear Model for Nuclear Power Plants using BDFs.............................................................................................. 132

Appendix B:Programs..........................................................................................135

Autobiography.................................................................................................... 173

## SUMMARY

Globally, digital control of nuclear reactors is finding acceptance around the academic and commercial circles at an increasing rate. Therefore, this resulted in the importance of controllers designed for nuclear plants using already existing control methods. State feedback controller offers much more benifits than a simple unitary output feedback controller. However there is an additional we need to estimate other states that are not available, hence problem becomes a synthesis of controller and estimator design. Optimal control design methodologies have been applied to nuclear power plants since 1970's. These and similar studies endeavor to increase controller robustness for some uncertainties which result from both measurement and inaccuracy in system modeling. Linear Quadratic Gaussian(LQG) accounts for a type of controller which is a synthesis of stochastic observer design (Kalman Filter) and Linear Quadratic Regulator(LQR) using the separation principle. These types of controllers are well studied , whose properties are well-known and their solutions formulated in the form of nonlinear matrix valued equations which are called Riccati differential Equations (RDEs). However solution of RDE and some nonlinear laws designed according to adaptive control are too computation-intensive for application to nuclear power plants. Due to non-linear mapping of Fuzzy Logic Controllers and Artificial Neural Networks(ANN) and also adaptivity properties of ANN, they have been employed in nuclear power plant control with an even increasing application rate.

In this spirit, I studied some controllers for nuclear power plants ranging from simple to more sophisticated and intelligent applications designed by the methodologies of fuzzy and neural networks.

# ÖZET

## YAPAY SİNİR AĞLARI ve BULANIK SİSTEMLERİN NÜKLEER GÜÇ SANTRALLARININ KONTROLUNDA KULLANILMASI

Bulanık mantık sistemleri ve sinir ağları konrol sistemlerinin zekasını artırmak için kullanılan yapay zeka araçlarıdır. Diğer taraftan nükleer reaktörler ancak liner olmayan sistemlerle matematiksel olarak ifade edilmektedirler. Zaman geçikmeli özellikleriyle kontrol uygulamaları için zor bir alanı oluşturmaktadırlar. Bununla birlikte kontrol tasarımlarının doğrudan birinin diğerine üstünlüğünü kanıtlamak zordur. Değişik kontrol tasarımlarının başlangıçta sayısal olarak simule edilerek nükleer reaktörler üzerinde denenmeleri gerekir. Karşılaştırılacak sonuçlara göre bu kontrol tasarımları nükleer reaktörler için değerlendirilebilir.

Nükleer reaktörlere uygulanan kontrol sistemlerini genel olarak yapılarına göre ikiye ayırmak mümkündür. Bunlar çıkış birim geri besleme kontrol sistemleri ve durum geri besleme kontrol sistemleridir. Bu kontrol sistemleri kullanımda oldukları uzun yıllar boyunca kendilerinden istenilenleri başarıyla yerine getirmişlerdir. Ancak temel olarak nükleer reaktörlerin en güvenli biçimde çalışması temel alındığından zamanın kontrol teknoloji de hesapa katılarak en basit yapıdaki kontrol sistemleri seçilmiş ve kullanılmıştır. Bununla beraber bilgisayar teknolojisindeki son gelişmeler daha karmaşık kontrol sistemlerinin nükleer reaktörlerde güvenli biçimde kullanılmasını olanaklı hale getirmiştir.

Sayısal bilgisayarların nükleer reaktörelerin kontrolünde kullanılması akademik ve ticari çevrelerde giderek artan bir oranda kabul görmektedir. Bundan dolayı, nükleer reaktörlerin daha iyi biçimde kontrol edilebilmesi için halihazırdaki kontrol tasarım metodlarının kullanılması önem kazanmıştır.

Durum geri besleme kontrol sisteminin basit bir çıkış birim geri besleme kontrol sistemine göre pek çok üstünlükleri vardır. Bunula birlikte, nükleer reaktörden ölçülemeyen durumlar için bir durum kestirici tasarlamak gerekmektedir. Bundan sonra problem bir durum kestirici ile denetleyicinin uygun biçimde birleştirilmesi olmaktadır.

Optimal Kontrol tasarım metodları 1970 'lerden beri nükleer güç reaktörlerine uygulanmaktadırlar. Bu ve benzeri çalışmalar, ölçmeden ve sistemi modellemedeki yanlışlıklardan en az etkilenecek, bu değişimlere en iyi biçimde tolere edebilecek denetleyiciyi tasarlamayı temel olarak amaç edinmişlerdir. Liner Karesel Gaussian, stokastik kestiriçi dizaynıyla (Kalman Süzgeçi) Doğrusal Karesel Düzenleyiçi dizaynını ayrılıklar prensibine göre birleştiren bir çesit kontrol sistemi tasarım metodudur. Bu tür tasarımlar iyi bilinen, özellikleri iyi çalışılmış ve bazılarına göre de artık eski moda tasarım metodlarıdır. Bu tür tasarım metodları sonuçta probleme göre lliner olmayan matris formunda matematiksel denklemeler olan Riccati Denklemleriyle çözülmektedir. Bununla beraber Ricatti Denklemlerinin çözümü ve uyarlamalı kontrole göre türetilebilecek liner olmayan kontrol kuralları nükleer reaktör kontrolu için oldukça hesaplama yoğun işlemler gerektirmektedir.

Bütün bu anlatılanlar ışığı altında bulanık mantık denetleyicileri ve sinir ağları liner olmayan sistemlerdeki uyarlama özelliklerinden dolayı nükleer güç santrallerinin kontrol sistemleri tasarımında gittikçe artan oranda kullanılmaktadırlar.

Genel olarak bunları gözeterek, en basitinden en karmaşık ve zekisine kadar nükleer güç santralleri için yapay sinir ağları ve bullanık mantık kullanılarak tasarlanan kontrol sistemlerini bu tezde araştırdım.

Nükleer güç ve suni zeka insanoğlunun en önemli başarılarıdır. Her ikisi de yaşadığımız son yüz yıl içinde düşünülmüş ve bazen sivil bazen de askeri amaçlar için kullanılmışlardır. Bunlardan ilkinin, toplumlar üzerindeki psikolojik etkisi büyük ölçüde soğuk savaş sırasındaki olası bir nükleer savaş felaketi yüzünden korkunç olmuştur. Diğer yandan yapay zeka son 10-15 seneye kadar bilim kurgu romanlarının en popüler konuları olarak kalmışlardır. Bu romanlarda anlatılan hikayeler büyük çoğunlukta insan benzeri robotların dünyadaki insan üstünlüğüne son verip insanları köle olarak kullanmalarıdır. Ne yazık ki suni zeka konusundaki araştırmacıların da belirtikleri gibi bu tip hikayeler alanın göstereceği gelişmelerde moral bozucu etkiler yapmıştır.

İlk nükleer güçle çalışan enerji santrallerinin kurulması ve bunların dünya çapında yagınlaşması nükleer enerjinin en ucuz ve insanoğlunun evrensel olarak elde edebileceği tek enerji kaynağı olduğunu göstermiştir.Genel olarak son yıllarda nükleer enerjiye olan talebin azalmasına rağmen uzayın derinliklerini keşfetmeyi düşleyen insanoğlu gelecek yüzyıllardaki diğer büyük projelerini gerçekleştirmek için nükleer enerjiyi kullanmalı ve onu araştırmalıdır. Bununla beraber günümüzdeki nükleer santrallerde karşılaşılan en önemli problem nükleer santrallerin güvenli biçimde çalıştırılmalarıdır. İkincisi ve belki en önemli problem nükleer atıklar sorunudur. Nükleer teknolojideki son gelişmeler nükleer atıklar sorununu çözebilir. Benzer olarak nükleer reaktörlerdeki yeni tasarım ve düzenlemeler reaktörlerin kararlılığını ve güvenilirliğini artırarak santrallerin daha güvenli biçimde çalışmalarını sağlayabilir.

Bütün bunlar ve sayısal bilgisayarlardaki son gelişmelerle beraber yeni veya oturmuş kontrol metodlarının nükleer reaktörlere uygulanması nükleer güç santrallerinin bakımı, verimliliği ve güvenliği açısından daha uygun çözümler üretebilecektir.

Araştırmalar kontrol kuramlarının güç santrallerine uygulanmasıyla ilgilidir. Genel olarak, nükleer reaktörleri modelleyen denklemler doğrusal olmayan ve zor (stiff) denklemlerdir. Ayrıca zaman geçikmeli özelliği ile nükleer reaktör dinamiği kontrol mühendisliği için meydan okuyucu araştırma konusunu oluşturmaktadır.

Bunlar dikkate alındığında, zeki denetleyicilerin diğerleri arasında aranan özellikleri sağlayabilecek görünen uygun adaylar olduğu anlaşılır. Yapay sinir ağları ve bulanık sistemler denetleyicilerin zekasını artırmada kullanılan etkili metodlardır. Fakat bu diğer klasik metodlarla tasarlanan kontrol sistemlerinin nükleer güç santrallarında güvenli ve yeterli biçimde kullanılmayacakları anlamına gelmez. Optimal kontrol uygulaması için gerekli olan Riccati Denklemlerinin çözülmesidir. Zamana göre değişen modellerde bu Riccati diferansiyel denklemlerinin çözümünü gerektirmektedir. Burda seçilen kontrol metodu kadar gerçek zamanlı nükleer reaktör kontrolunda kullanılan çözüm algoritmasının da önemi ortaya çıkmaktadır. Hatta bazı algoritmalar için çözüm olanaksızlıklaşmakta ya da büyük bir hatayla bulunmaktadır. Bu da sistemin kararsızlığına yol açmak gibi nükleer santrallerin çalışmasında son derece tehlikeli durumların ortaya çıkmasına neden olmaktadır.

Bu ve benzeri sebepler uyarlamalı kontrolün nükleer reaktörlere güvenli biçimde uygulanmasını güçleştirmektedir. Lineer karesel optimal kontrol problemlerinde liner modeller kullanılmaktadır. Genel olarak nükleer reaktörler için türetilen liner olmayan modeller bir çalışma noktasında lineerleştirilmekte ve bu lineerleştirilmiş modeller parametreleri zamana göre değişmeyen denetleyiciler üretmektedir. Ancak kullanılan lineer modeller reaktörlerin doğal lineer olmayan özelliklerini tam olarak karşılamamakta, kontrol sistemi tasarımında başlangıçta kabul edilen bir hata olmaktadırlar. Bu sorunun üstesinden gelebilmek için reaktörün çeşitli çalışma bölgeleri için lineerleştirme yapılmakta ve her bölge için farklı denetleyeçi parametreleri hesaplanmaktadır.

Yapay sinir ağları ve bulanık mantık denetleyicilerinin önemi burda ortaya çıkmaktadır. Genel olarak uygulamalarda farklı çalışma bölgeleri için tasarlanmış denetleyicinin kontrol sinyalleri, yapay sinir ağları ya da otamatik olarak kuralları ayarlanabilen bulanık mantık kontrol sistemlerine örnek, öğrenilmesi istenilen bilgi olarak girilir. Bu yapay sinir ağları ve bulanık mantık sistemlerinin doğrudan kontrolu uygulamasına örnek verilebilir. Diğer taraftan nükleer reaktörlerin dinamiği bu sistemler tarafından öğrenilebilir. Bu da sistemin durumlarının daha kesin biçimde belirlenmesini sağlayan sistemlerin tasarımında kullanılabilir.

Bu tez kapsamı içinde klasik en basit yapıdaki denetleyiciler başlamak üzere en karmaşığına doğru yapay sinir ağları ve bulanık mantık sistemlerini kapsayacak biçimde kontrol sistemlerinin tasarımları yapılmış ve bunlar nükleer reaktörlerin liner olmayan modelleri kullanılarak bilgisayarla simule edilmişlerdir.

## List of Figures

| | |
|---|---|
| Fig 1.1 | Structural overview of non-intelligent controllers |
| Fig 1.2 | Structural overview of intelligent controllers |
| Fig 2.5.1 | Classical control of nuclear power plants |
| Fig 2.5.2 | Root locus of linearized model with no temperature feedback |
| Fig 2.5.3 | Root locus of linearized model with temperature feedback |
| Fig 2.5.1.1 | Dominat eigenvalues of linearized sytem without temperature feedback |
| Fig 2.5.1.2 | Dominat eigenvalues of linearized sytem with temperature feedback |
| Fig 2.5.1.3 | Simulation results for nonlinear and linearized models |
| Fig 2.5.2.1 | Simulation results for COC without temperature feedback |
| Fig 2.5.2.2 | Simulation results for COC with temperature feedback |
| Fig 2.6.3.1 | State feedback control with state estimation for nuclear power plants |
| Fig 2.6.3.2 | Simulation results for SFCSE |
| Fig 3.5.1 | Simulation results for optimal control |
| Fig 4.2.1 | Discrete-time Kalman filter |
| Fig 4.2.2 | Continuos -time Kalman filter |
| Fig 4.3.1 | Block diagram classical output control with Kalman filter |
| Fig 4.3.2 | Simulation results for COC with Kalman filter |
| Fig 4.5.1 | Simulation results for LQG |
| Fig 5.3.1 | Basic configuration of fuzzy logic systems |
| Fig 5.3.2 | Fuzzy Logic Control of Nuclear Power Plant |
| Fig 5.6.4.1 | Supervisory signals produced by LQG |
| Fig 5.7.1 | Simulation results for an automatically tuned fuzzy logic controller |
| Fig 5.8.1 | Control Surface of FLC |
| Fig 6.2.1 | Neural Networks and Learning Algorithms |
| Fig 6.3.1 | Neuron |
| Fig 6.3.2 | Feedforward multi-layered neural network |
| Fig 6.8.1 | Model reference adaptive control(MRAC) |
| Fig 6.8.2 | Self-tuning controller(STR) |
| Fig 6.8.3 | Direct adaptive control of nonlinear plants using neural networks |
| Fig 6.8.4 | Neural networks in system identification |

Chapter one

# OVERVIEW

## 1.1. Motivation

Nuclear power and artificial intelligence are the most challenging accomplishments of mankind . Both of them are conceived and engineered in the present century for civilian and military purposes. Psychological impact of the first one on public security was terrible (mostly due to catastrophic results of a possible nuclear disease during cold war). On the other hand, artificial intelligence was the popular subject in science-fiction novels until 10-15 years ago (in a typical story, human-like robots terminate human superiority on the earth and kill or use them as their slaves; unfortunately it was reported by artificial intelligence researchers that the effect of these stories was discouraging for the development of the field)[27]. But with the construction of the first nuclear power plant for energy production and its expansion around the world, it is understood that nuclear energy is the cheapest and unique energy which mankind may handle universally. In spite of the general decreasing trend in the demand for nuclear energy, ultimately, mankind who dreams to discover the depths of space should search and use the nuclear energy for his other challenging projects in the next centuries[82]. However, main problems arising from nuclear power plants are their safe and reliable operation and secondly the management of nuclear waste. Any way, the main reason for the decrease in the demand for nuclear energy is mostly due to public unconfidence in nuclear power nuclear technologies, especially in fuel preparation, may solve nuclear waste problem and similarly new design and modifications in nuclear reactors may improve the stability and reliability of the plants to enhance the security. However, developing and applying new and embedded control strategies with the last developments in digital control technology will produce more suitable solutions to improve security, efficiency, maintainability of nuclear power plants [2], [3].

Researchers in this field searched applicability of control theories to power plants. As a general statement , mathematical model of nuclear plants are given by non-linear and stiff equations. Also dynamic time-delay properties of nuclear reactors provide another challenging research topic for control engineers. Among them, intelligent controllers are good candidates to satisfy required qualifications sought for a nuclear power plant such as robustness, stability and reliability.

Neural networks and fuzzy systems are employed as a powerful tools to increase the intelligence of controllers. But this does not mean that classical control philosophies such as stochastic optimal control can not be embedded in a suitable and reliable operation. In the case of optimal control, the price paid for optimal controller is to find the solutions of Algebraic Riccati Equations (ARE), sometimes Riccati Differential Equations (RDE). Generally stiffness in these algorithms (solver for RDE and ARE) is a major problem. Therefore, not only designing system using optimal control concept but chosen algorithms play a significant role in the system stability. Error analysis in the solutions of equations (proposed by optimal control or other design methodologies) should be taken into consideration for good system reliability. The word, reliability used here refers to numerically solvability of these equations by algorithms and their failure mechanism in numeric computation. It is mostly due to highly stiff equations arising in this field. This stiffness results in some computational difficulties for some algorithms. Any way main restriction on the applications of adaptive systems is that their solutions are not guarantied for a reliable operation. The time-invariant models provide time-invariant controller gains that are solutions of ARE in the linear quadratic optimal control. However time-invariant models do not match with the physical reality because of general nonlinearity of plant and some other physical events that affect the process such as burn-up.

The significance of fuzzy logic controllers and neural networks appears at this point is that their non-linear mapping and learning abilities reported in numerous studies and applications, may be used to overcome the problems due to nonlinearity and physical events during the operation of nuclear power plants.

## 1.2 Objective

Objective of this work cah be summarized in the following items;

1- Understanding the dynamics of nuclear plants and defining problems resulting from plants for controller design.

2- Surveying already designed or existing controllers for nuclear plants such as classical output controllers, state feedback controllers with pole placements, LQ controllers, and defining problems with these controllers.

3- Designing intelligent controllers using fuzzy logic and neural network methodologies.

4- Supplying sufficient proofs for intelligent controllers that they can be employed confidentially in the control of nuclear power plants.

5- Providing the mathematical fundamentals and necessary computational tools to solve the problems for controller design.

.6- Providing computer aided simulations for the experimental controller implementations in nuclear plants.

## 1.3 Control Methodologies

We will start the work classifying the control methodologies into non-intelligent and intelligent controllers. Fig 1.1 illustrates the structural overview of some non-intelligent controllers. Unitary output feedback controllers, dynamic compensators (PI, PID) and state feedback controllers may be accounted in this group. Classical stability analysis such as root locus, pole placement techniques and deterministic linear observer designs are typical methods in the design of controllers. we can improve controller in dynamic response and robustness using the optimal control theory. Linear quadratic (LQ) regulator is a simplest example in optimal controller design. Using stochastic observer (Kalman filter) with LQ, we can improve the robustness of controller for disturbances in plant.

Fig 1.2 summarizes the intelligent control with neural networks and fuzzy systems.As a general statement, intelligent controllers are not model-based controllers. They can be applied to any nonlinear model. Among them, fuzzy logic controllers(FLCs) are knowledge-based systems. Previously obtained experiences about a process can be used in the intelligent control design. FLC is a computational tool to formulate the experiences in linguistic forms. Neural networks are powerfull tools using in intelligent control because of their learning abilities and flexibility.

## 1.4. Organization

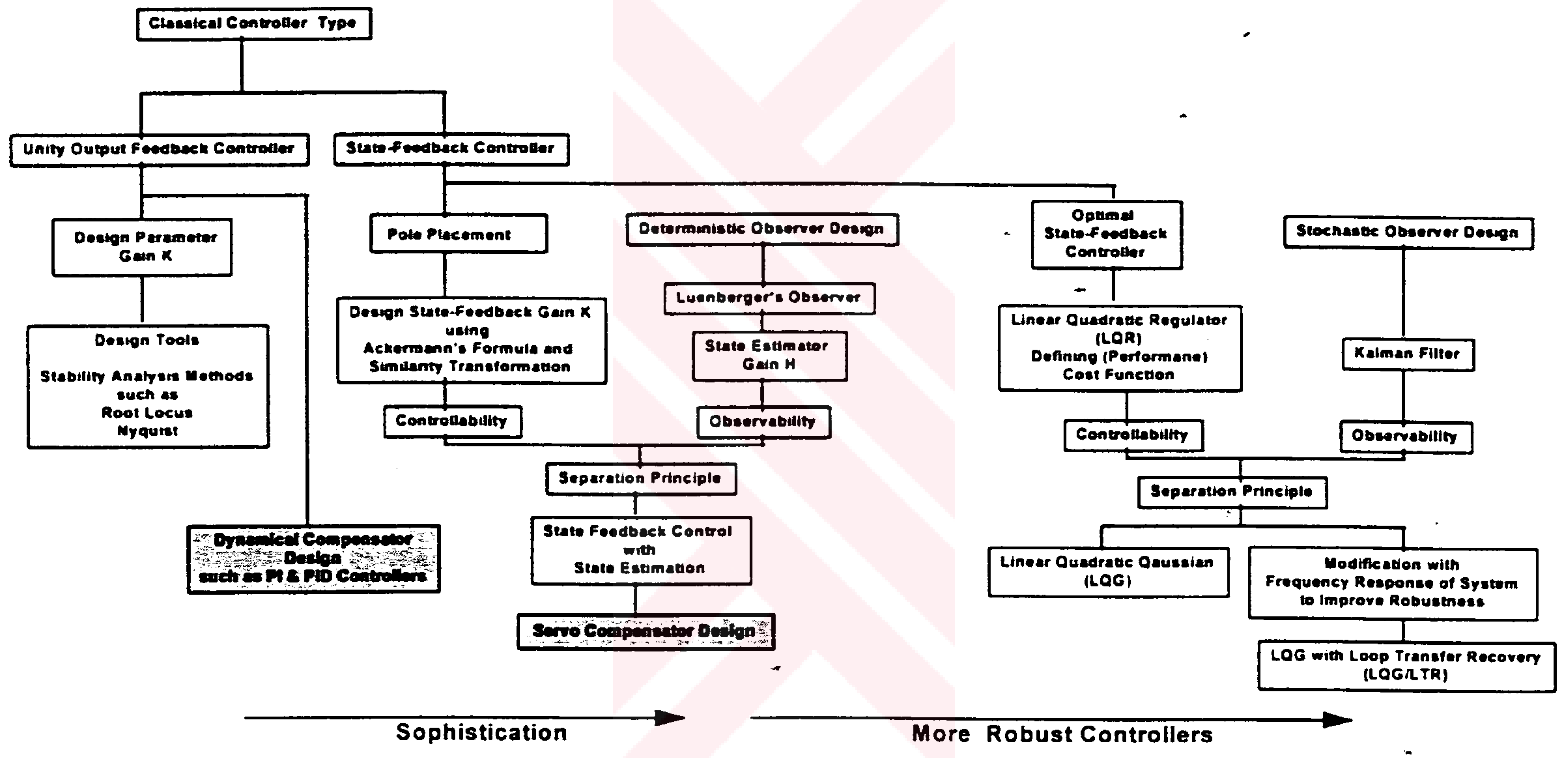
Chapter Two is devoted to dynamics of nuclear plants and conventional control methods. Classical output control, state feedback control with state estimation using pole placement techniques are presented with the explanatory results of the inherent nonlinearity of plants. A state feedback controller is designed with a state observer (Luenberger's observer) and results of simulation with nonlinear plant are presented in this chapter. we will concentrate the nonlinearity and stability of the plants applying the several theorems, such as linearization and root locus analysis. Also we will mention about the stiffness arising in the mathematical model of plants.

Chapter Three is devoted to optimal control, optimal state-feedback control has been designed for nuclear power plants and simulation results are presented at the end of the chapter. With this chapter, we will introduce the optimality criteria to minimize a performance index defined for time -invariant or time varying linearized models of nuclear plants, using the linear quadratic methods. Also a numerical example related with stiffness arising in the optimal control of nuclear plants will be presented.

Chapter Four is organized for stochastic optimal control results of simulation with nonlinear model of nuclear power plant. With simulation results of nonlinear plant for a Pressurized Water Reactor(PWR) a stochastic problem for white, gaussian, zero mean noise has been solved using a scalar Kalman Filter. LQG design has been discussed in this chapter.We will presents the results of the stochastic optimal estimator (Kalman Filter) for plants. With this chapter, we will also have introduced the methods to improve the robustness of controllers for nuclear power plants.

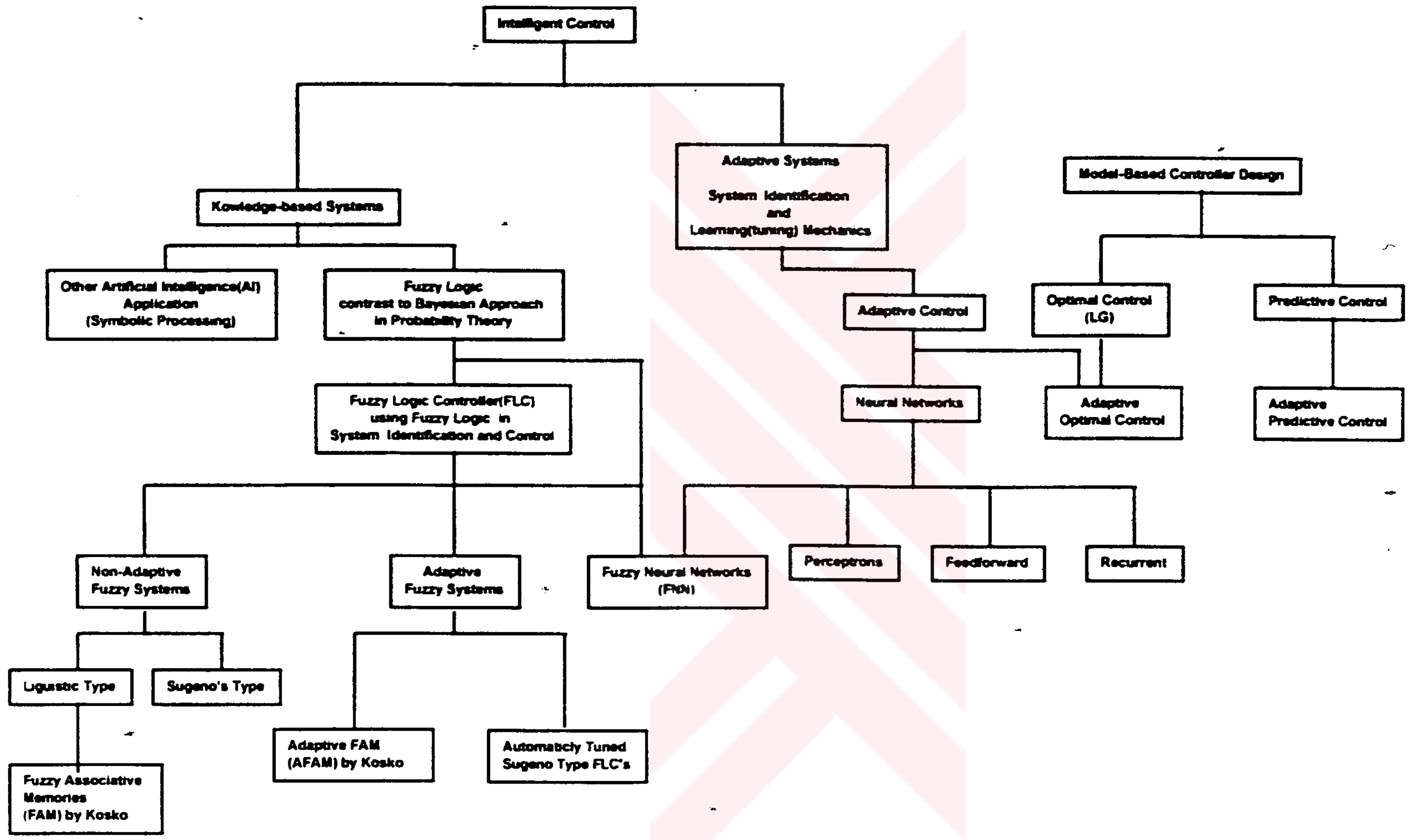
Chapter Five presents the results of an automatically tuned fuzzy logic controller. Tuning mechanism has been constructed using both least square estimation and mean square estimation (Kalman Filter) with an optimal controller output as supervisory signal to tune fuzzy rules properly. With fuzzy logic controllers, we will introduce the information-based systems, human-like thinking and intelligent control.

In Chapter six, we will introduce the neural network methodologies to develop controllers for nuclear power plants. A direct control of nuclear plant using neural networks will presented in this chapter. Learning ability, flexibility nonlinear mapping capability of neural networks will be examined in an application of neurocontroller which will be designed in this chapter.



Classical Controller Type
Unity Output Feedback Controller
State-Feedback Controller
Design Parameter Gain K
Design Tools
Stability Analysis Methods such as Root Locus Nyquist
Dynamical Compensator Design such as PI & PID Controllers
Pole Placement
Design State-Feedback Gain K using Ackermann's Formula and Similarity Transformation
Controllability
Deterministic Observer Design
Luenberger's Observer
State Estimator Gain H
Observability
Separation Principle
State Feedback Control with State Estimation
Servo Compensator Design
Optimal State-Feedback Controller
Linear Quadratic Regulator (LQR) Defining (Performane) Cost Function
Controllability
Stochastic Observer Design
Kalman Filter
Observability
Separation Principle
Linear Quadratic Qaussian (LQG)
Modification with Frequency Response of System to Improve Robustness
LQG with Loop Transfer Recovery (LQG/LTR)
Sophistication
More Robust Controllers

Fig. 1.1 Structural Overview of non-intelligent Controllers.
(Shaded areas are considered in this work)



Intelligent Control
Kowledge-based Systems
Adaptive Systems System Identification and Learning(tuning) Mechanics
Model-Based Controller Design
Other Artificial Intelligence(AI) Application (Symbolic Processing)
Fuzzy Logic contrast to Bayesian Approach in Probability Theory
Adaptive Control
Optimal Control (LG)
Predictive Control
Fuzzy Logic Controller(FLC) using Fuzzy Logic in System Identification and Control
Neural Networks
Adaptive Optimal Control
Adaptive Predictive Control
Non-Adaptive Fuzzy Systems
Adaptive Fuzzy Systems
Fuzzy Neural Networks (FNN)
Perceptrons
Feedforward
Recurrent
Liguistic Type
Sugeno's Type
Adaptive FAM (AFAM) by Kosko
Automaticly Tuned Sugeno Type FLC's
Fuzzy Associative Memories (FAM) by Kosko

Fig 1.2 Structural overview of intelligent controllers

CHAPTER TWO

# DYNAMICS OF NUCLEAR POWER PLANTS AND CLASSICAL CONTROL METHODS

In this chapter, the derivation of a mathematical model for a nuclear power plant is presented. PWR is selected for nuclear plant and it's lumped circuit parameters are included for the derivation of mathematical model. Unless stated otherwise, the same PWR model is used to develop controllers throughout this work.

## 2.1. Nomenclature

| | |
|---|---|
| $n$ | $\equiv$ neutron density (n/cm$^3$) |
| $n_r$ | $\equiv n/n_o$ neutron density relative to density at rated power. |
| $n_o$ | $\equiv$ neutron density at rated power. |
| $G$ | $\equiv$ number of delayed neutron groups. |
| $c_i$ | $\equiv$ density of delayed neutron precursor group i (atom per cm$^3$). |
| $c_{ri}$ | $\equiv c_i/c_{io}$ relative precursor group i density |
| $c_o$ | $\equiv$ initial equilibrium precursor density |
| $\delta\rho \equiv \frac{k-1}{k} = \frac{\Delta k}{k}$ | $\equiv$ reactivity |
| $\delta\rho_r$ | $\equiv$ reactivity due to the control rod. |
| $\lambda$ | $\equiv$ effective precursor radioactive decay constant (s$^{-1}$). |
| $\lambda_i$ | $\equiv$ radioactive decay constant(s$^{-1}$) of delayed neutron precursor group i. |
| $\Lambda$ | $\equiv$ effective prompt neutron lifetime (s). |

$\beta_i \equiv$ fraction of delayed fission neutrons in group i.

$\beta \equiv \sum_{i=1}^{G} \beta_i$

$f_f \equiv$ fraction of reactor power deposited in the fuel.

$\mu_f \equiv$ total heat capacity of the fuel and structural material (MWsec/°C)

$\mu_c \equiv$ total heat capacity of the reactor coolant (MWsec/°C).

$\Omega \equiv$ heat transfer coefficient between fuel and coolant (MW/°C)

$P_{0a} \equiv$ reactor initial equilibrium power level (MW).

$P_a \equiv$ reactor power (MW).

$M \equiv$ mass flow rate times heat capacity of the coolant (MW/°C).

$z_r \equiv$ control rod speed as control input (fraction of core length per second).

$G_r \equiv$ total reactivity worth of the rod.

$\alpha_f \equiv$ fuel temperature reactivity coefficient.

$\alpha_c \equiv$ coolant temperature reactivity coefficient.

$T_f \equiv$ average reactor fuel temperature(°C).

$T_l \equiv$ temperature of the coolant leaving the reactor (°C).

$T_e \equiv$ temperature of the coolant entering the reactor (°C).

$T_c \equiv$ average reactor coolant (water) temperature ( $T_l + T_e$) / 2.

$T_{f0} \equiv$ initial steady-state temperature of the fuel (°C).

$T_{l0} \equiv$ initial steady-state temperature of the coolant leaving (°C).

$T_{e0} \equiv$ initial steady-state temperature of the coolant entering the reactor (°C).

$T_{c0} \equiv$ initial steady-state value of the coolant temperature (°C).

## 2.2. Simulation Model

The mathematical model of a pressurized water reactor(PWR) is derived to discuss the responses of controller investigated in this work following the lines of [1]. While keeping the main derivation concepts unchanged some parameters were simulated making them power level dependent to match more actual plant dynamics with simulation model throughout this work. Main dynamics of nuclear reactor is represented by point kinetics equations with G groups of delayed neutron precursors. For simplicity, one delayed neutron group was used in the controller design stage and related analysis. The lumped parameter model was used for the fuel and coolant temperature calculations.

Simply, point kinetics equations with G groups of delayed neutron precursors:

$$\frac{dn}{dt} = \frac{\delta\rho - \beta}{\Lambda} n + \lambda_i c_i \qquad (2.2.1)$$

$$\frac{dc_i}{dt} = \frac{\beta_i}{\Lambda} n - \lambda_i c_i \qquad i=1.....G, \qquad (2.2.2)$$

Following normalized versions of equations (2.2.1) and (2.2.2) are more convenient to use for computational purpose :

$$\frac{dn_r}{dt} = \frac{\delta\rho - \beta}{\Lambda} n_r + \frac{1}{\Lambda}\sum_{i=1}^{G} \beta_i c_{ri} \qquad (2.2.3)$$

$$\frac{dc_{ri}}{dt} = \lambda_i n_r - \lambda_i c_{ri} \qquad i=1........G, \qquad (2.2.4)$$

For one delayed neutron group, normalized point kinetics equations are

$$\frac{dn_r}{dt} = \frac{\delta\rho - \beta}{\Lambda} n_r + \frac{\beta}{\Lambda} c_r \qquad (2.2.5)$$

$$\frac{dc_r}{dt} = \lambda n_r - \lambda c_r \qquad (2.2.6)$$

We can formulate power transferred from fuel to coolant in MW;

$$P_c = \Omega\,(T_f - T_c) \qquad (2.2.7)$$

and power removed from coolant in MW;

$$P_e = M\,(T_l - T_c) \qquad (2.2.8)$$

The differential equations governing the fuel and coolant temperatures using lumped parameters;

$$f_r P_a(t) = \mu_f \frac{d}{dt} T_f + P_c(t) \qquad (2.2.9)$$

$$(1 - f_r)P_a(t) + P_c(t) = \mu_c \frac{d}{dt} T_l + P_e(t) \qquad (2.2.10)$$

where reactor power at time t, $P_a(t) = P_{0a}(t) n_r(t)$.

Reactivity entered due to control rod is represented by the following differential equation:

$$\frac{d\delta\rho_r}{dt} = G_r z_r \qquad (2.2.11)$$

Total reactivity input to the points kinetics equations is the summation of control rod reactivity and temperature feedback reactivities of the fuel and coolant;

$$\delta\rho = \delta\rho_r + \alpha_f (T_f - T_{fo}) + \alpha_c (T_c - T_{co}) \qquad (2.2.12)$$

After necessary manipulations, we obtain the following 5th order non-linear differential equation set that is based on point kinetics and a lumped parameter thermal-hydraulic model governing the dynamics of nuclear plant.

$$\frac{dn_r}{dt} = \frac{\delta\rho - \beta}{\Lambda} n_r + \frac{1}{\Lambda}\sum_{i=1}^{G}\beta_i c_{ri} \tag{2.2.13}$$

$$\frac{dc_{ri}}{dt} = \lambda_i n_r - \lambda_i c_{ri} \qquad i=1........G. \tag{2.2.14}$$

$$\frac{dT_f}{dt} = \frac{f_f P_{oa}}{\mu_f} n_r - \frac{\Omega}{\mu_f} T_f + \frac{\Omega}{2\mu_f} T_l + \frac{\Omega}{2\mu_f} T_e \tag{2.2.15}$$

$$\frac{dT_l}{dt} = \frac{(1-f_f)P_{oa}}{\mu_c} n_r + \frac{\Omega}{\mu_c} T_f - \frac{(2M-\Omega)}{2\mu_c} T_l + \frac{(2M-\Omega)}{2\mu_c} T_e \tag{2.2.16}$$

$$\frac{d\delta\rho_r}{dt} = G_r z_r \tag{2.2.17}$$

where $\delta\rho$ is given by Eq.(2.2.12).

The proposed differential equations are nonlinear because of $\delta\rho n_r$ product.

## 2.3. Linearization of Nonlinear Systems

Most of the applications are related to linear system model. The linearization of a nonlinear system is a powerful nonlinear system analysis tool. Main theorem is based on Taylor's expansion of a nonlinear function at an operating point [4]. The system is assumed as a linear system at the vicinity of this operating point and all the linear system analysis tools become applicable to nonlinear system.

Theorem 2.3.1

Let's assume that we have a nonlinear differential equation set expressed in the general vector form as follow;

$$\dot{x} = f(x,u) \quad , \qquad f=[f_1\ f_2\ ...f_n]^t \tag{2.3.1}$$

These equations may be rewritten in the following linear state-space form:

$$\dot{x} = Ax + Bu \tag{2.3.2}$$

where A and B are the proper Jacobian matrices;

$$A = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdot\ \cdot & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdot\ \cdot & \frac{\partial f_2}{\partial x_n} \\ \cdot & \cdot & \cdot\ \cdot & \cdot \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdot\ \cdot & \frac{\partial f_n}{\partial x_n} \end{bmatrix} = \left.\frac{\partial f}{\partial x}\right|_{x^\circ,u^\circ} \qquad (2.3.3)$$

$$B = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} & \cdot\ \cdot & \frac{\partial f_1}{\partial u_r} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} & \cdot\ \cdot & \frac{\partial f_2}{\partial u_r} \\ \cdot & \cdot & \cdot\ \cdot & \cdot \\ \frac{\partial f_n}{\partial u_1} & \frac{\partial f_n}{\partial u_2} & \cdot\ \cdot & \frac{\partial f_n}{\partial u_r} \end{bmatrix} = \left.\frac{\partial f}{\partial u}\right|_{x^\circ,u^\circ} \qquad (2.3.4)$$

Let $x^0$ and $u^0$ be the operation point of the nth order nonlinear system and the input at the system operation at $x^0$.

A perturbation is defined around the operating point;

$$u = u^0 + \delta u \qquad (2.3.5)$$

$$x = x^0 + \delta x \qquad (2.3.6)$$

From (2.3.1),

$$\frac{d}{dt}(x^\circ + \delta x) = \dot{x}^\circ + \delta\dot{x} = f(x^\circ + \delta x, u^\circ + \delta u) \qquad (2.3.7)$$

Using Taylor series expansion for the jth nonlinear equation;

$$\begin{aligned} \dot{x}_j^\circ + \delta\dot{x}_j = f_j(x^\circ,u^\circ) + \left.\frac{\partial f_j}{\partial x_1}\right|_{x^\circ,u^\circ} \delta x_1 + \cdots\cdots + \left.\frac{\partial f_j}{\partial x_n}\right|_{x^\circ,u^\circ} \delta x_n \\ + \left.\frac{\partial f_j}{\partial u_1}\right|_{x^\circ,u^\circ} \delta u_1 + \cdots\cdots + \left.\frac{\partial f_j}{\partial u_r}\right|_{x^\circ,u^\circ} \delta u_r \end{aligned} \qquad (2.3.8)$$

and from main definition of nonlinear equations (2.3.1);

$$\dot{x}_j^\circ = f_j(x^\circ,u^\circ) \qquad (2.3.9)$$

following equation is obtained as a result:

$$\delta\dot{x}_i = \left.\frac{\partial f_i}{\partial x_1}\right|_{x^\circ,u^\circ}\delta x_1 + \cdots\cdots + \left.\frac{\partial f_i}{\partial x_n}\right|_{x^\circ,u^\circ}\delta x_n + \left.\frac{\partial f_i}{\partial u_1}\right|_{x^\circ,u^\circ}\delta u_1 + \cdots\cdots + \left.\frac{\partial f_i}{\partial u_r}\right|_{x^\circ,u^\circ}\delta u_r \tag{2.3.10}$$

we can express (2.3.10) in the form of (2.3.2);

$$\delta\dot{x} = A\delta x + B\delta u \tag{2.3.11}$$

where A and B are the Jacobian matrices that are defined in (2.3.3) and (2.3.4).

## 2.4. Linearized Model of Nuclear Plants

In this section, linearization theorem that explained in section (2.3.) will be applied to a physical system, nuclear power plant whose model is developed in the previous sections.

Nonlinear differential equations including (2.2.5), (2.2.6), (2.2.15), (2.2.16) and (2.2.17) are linearized and resultant linear form is written as [1];

$$\dot{x} = Ax + Bu\ , \tag{2.4.1}$$

$$y = C\,x + Du \tag{2.4.2}$$

State vector : $x = \begin{bmatrix} \delta n_r \\ \delta c_r \\ \delta T_f \\ \delta T_l \\ \delta \rho_r \end{bmatrix}$ , control input $u=[z_r]$ , system output $y=[\delta n_r]$

(2.4.4)

$$\text{system matrix;}A = \begin{bmatrix} -\beta/\Lambda & \beta/\Lambda & n_{ro}\alpha_f/\Lambda & n_{ro}\alpha_c/2\Lambda & n_{ro}/\Lambda \\ \lambda & -\Lambda & 0 & 0 & 0 \\ f_f P_o/\mu_f & 0 & -\Omega/\mu_f & \Omega/2\mu_f & 0 \\ (1-f_f)P_o/\mu_c & 0 & \Omega/\mu_c & -(2M+\Omega)/2\mu_c & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(2.4.5)

$$\text{input matrix; } B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ G_r \end{bmatrix}, \text{ output matrix } C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.4.6}$$

where A and B are proper Jacobian matrices. It is also important to point out that products of second order deviations, such as $\delta\rho_r$ times $\delta n_r$ are ignored. In the state equations, matrix D represents direct coupling between input and output with a proper size. It is assumed that coupling matrix D has no influence on our calculations and it is ignored .

### 2.4.1. System Parameters Dependency on $n_{r0}$

Besides the system matrix A dependency on equilibrium neutron density $n_{r0}$, to improve the simulation model accuracy, also parameters $\alpha_f$, $\alpha_c$, $\mu_c$, $\Omega$ and M are dependent on $n_{r0}$. For parameter's dependency on $n_{r0}$, see [46][47].

## 2.5. Classical Output Control (COC)

Classical control approach is the most embedded applications in reactor control. Fig 2.5.1 shows this kind of control with single input; control rod speed, single output; neutron density (power of plant). As it is seen from figure, gain $G_C$ is the only design parameter of control system. Output power of plant is compared with demand power signal $P_d$ and amplified error signal is input to control rod speed $z_r$.

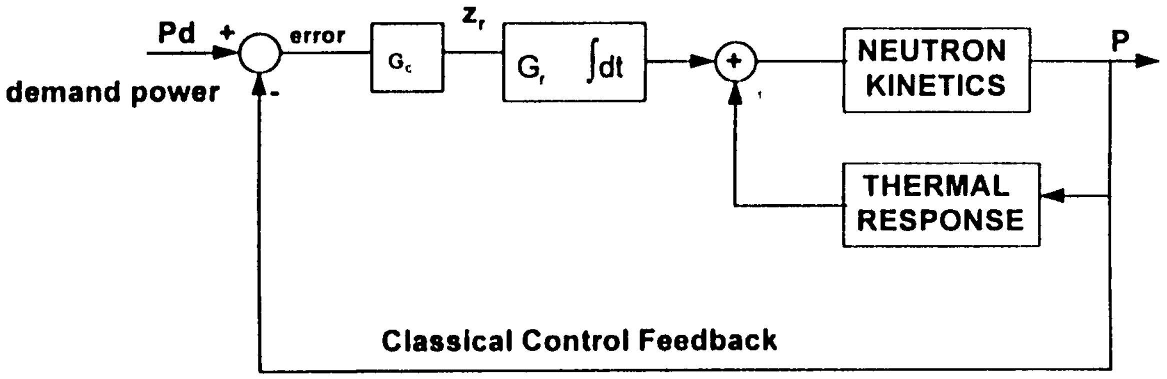


Fig 2.5.1 Classical control of nuclear reactors with single input single output and gain $G_C$.

The only design parameter gain $G_c$ will be selected using root locus analysis. To do this, required open-loop transfer function may be found using several tools. One way is to use the ss2zp.m Matlab file from it's control toolbox. We can find it's transfer function from control canonical state-space form of linearized model. Required analysis tool used by ss2zp or in finding control canonical form, is the similarity transformation that plays a significant role in the linear system theory. The open-loop transfer function of linearized model at initially equilibrium conditions ($n_{r0}$=1) with no temperature feedback is

$$g(s) = \frac{n_r(s)}{z_r(s)} = \frac{(G_r/\Lambda)(s+\lambda)}{s^2(s+\lambda+\beta/\Lambda)} \qquad (2.5.1)$$

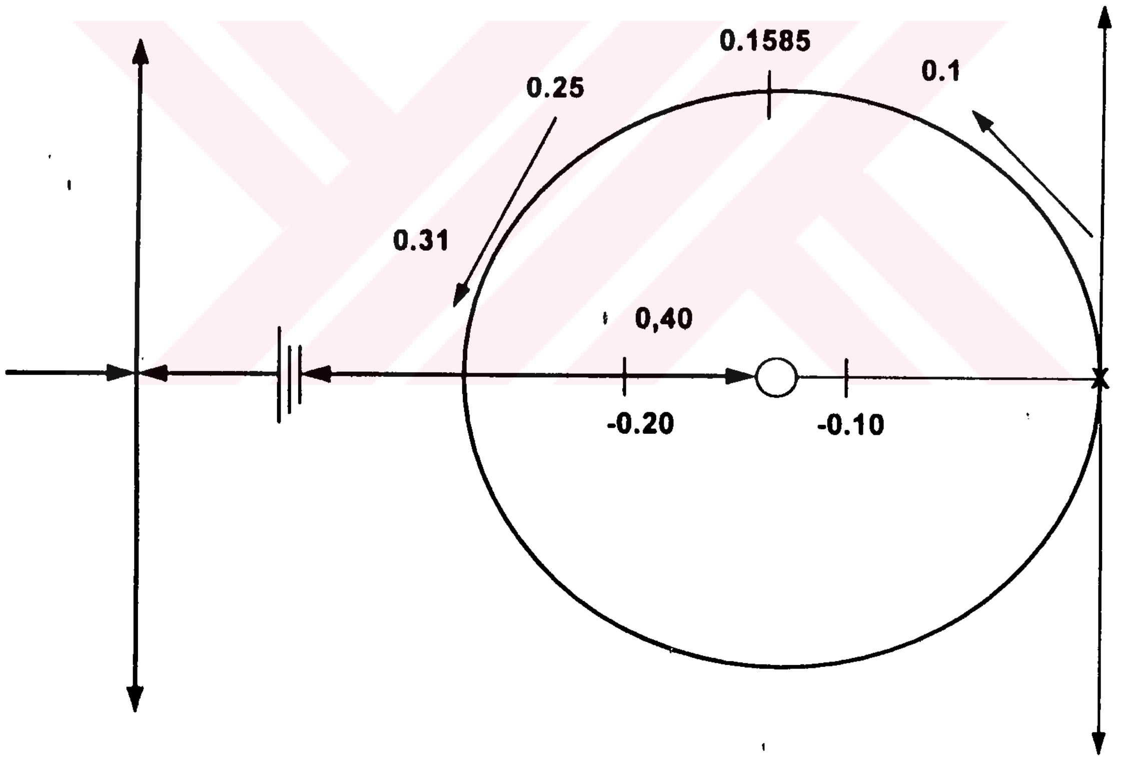


Fig 2.5.2 Root locus plot of linearized model with no temperature feedback at start-up.

From root locus plot , at $G_c$= 0.1585 system response satisfies damping ratio of 0.70 for a step input to second order system. In fact, since our system is fifth order, dominant eigenvalue may be used as poles of second order system to carry out system responses analysis properly for higher order systems. In our case dominant eigenvalue of close loop system is the largest eigenvalue. Poles of close loop system at $G_c$=0.1585 are -64.8812 and dominant poles ; -0.1219±0.1252j. General second order system is

$$G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$ where $\xi$ and $\omega_n$ are damping ratio and natural frequency, respectively. For $s_{1,2}$=-0.1219±j 0.1252 $\omega_n$=0.175 and $\xi$=0.69.

Transfer function of open-loop system with temperature feedback is

$$g(s) = \frac{n_r(s)}{z_r(s)} = \frac{(G_r/\Lambda)(s+\lambda)(s+0.2381)(s+1.373)}{s(s+0.1052)(s+0.8332)(s+1.397)(s+64.40)} \qquad (2.5.2)$$

Fig 2.5.3 illustrates the root locus of linearized model with temperature feedback at start-up. For a good system response, Gc=0.5 from root-locus. In this case, dominant eigenvalue of close loop system is -0.1194±j 0.0358 with a damping ratio of 0.96 . The damping ratio 0.96 refers to more sluggish response than response of a model without temperature feedback . Since it is impossible to improve the response for output feedback control, it appears as a main restriction on this kind of control strategy.

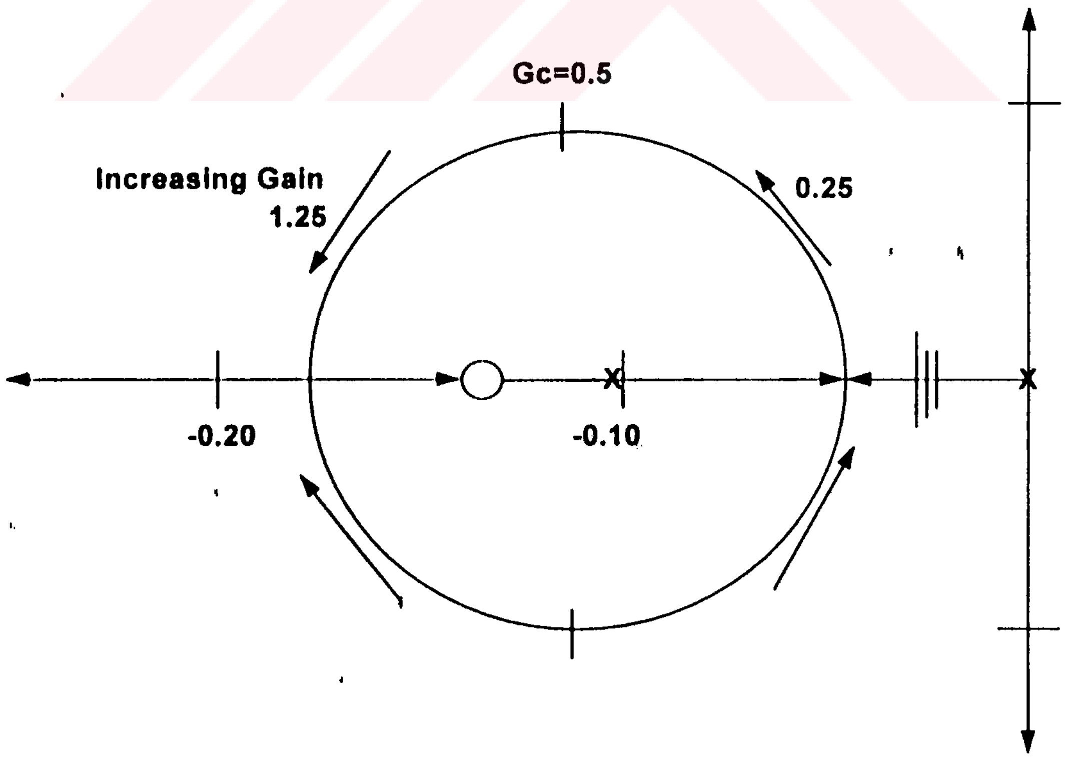


Fig 2.5.3 Root locus of linearized model with temperature feedback

### 2.5.1. Inherent Nonlinearity of Plants

In this section, the effect of nolinearity arising in state equations of system, and accuracy of linearized model will be investigated. Fig 2.5.1.1 and Fig 2.5.1.2 illustrate the dominant eigenvalues of close loop system for linearized model at different loop gains and different operating points from 10% power to 150% power with 10% equal incremental steps in power level.

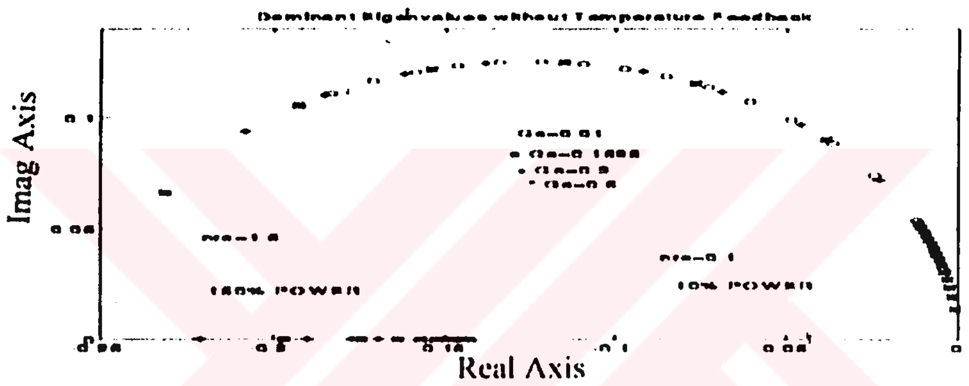


Fig 2.5.1.1. Dominant eigenvalues of linearized system(without temperature feedback) at operating points from 1% power to 150% power with 10% equal increase steps for loop gains Gc=0.01, Gc=0.1585, Gc=0.3, Gc=0.5.

To examine the influence of nonlinearity on state variables, simulation results of nonlinear model and linearized model at start-up are compared in Fig 2.5.1.3 for relative power, relative precursor density, fuel temperature, reactor exit temperature and reactivity (90% power drop is entered to reactor as power demand ).

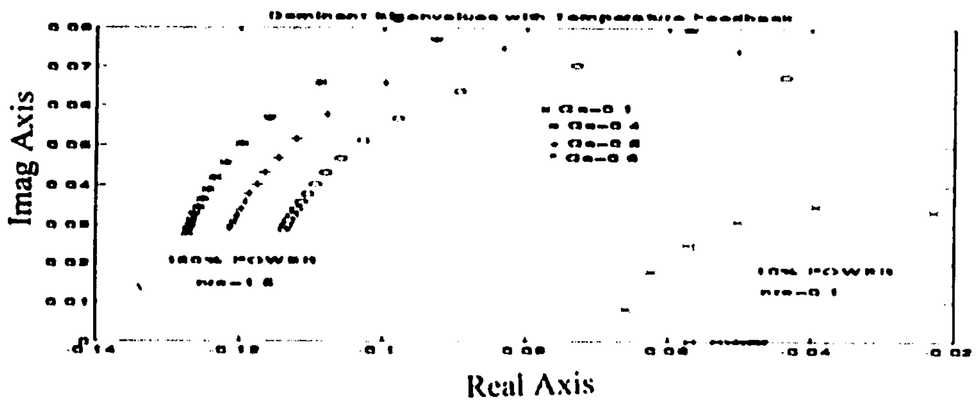
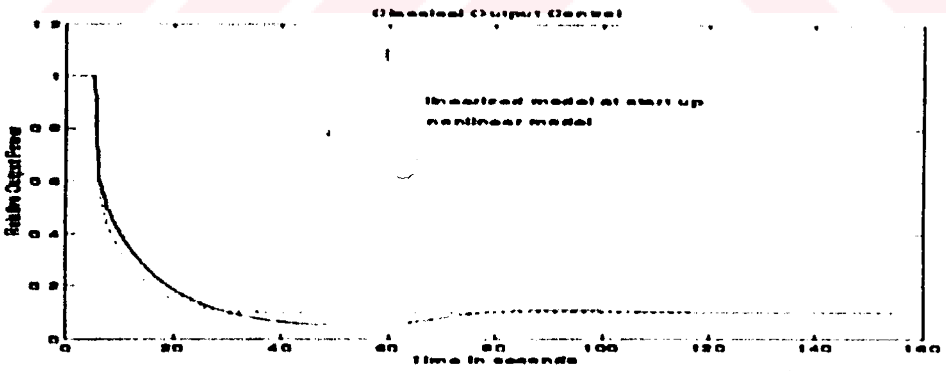
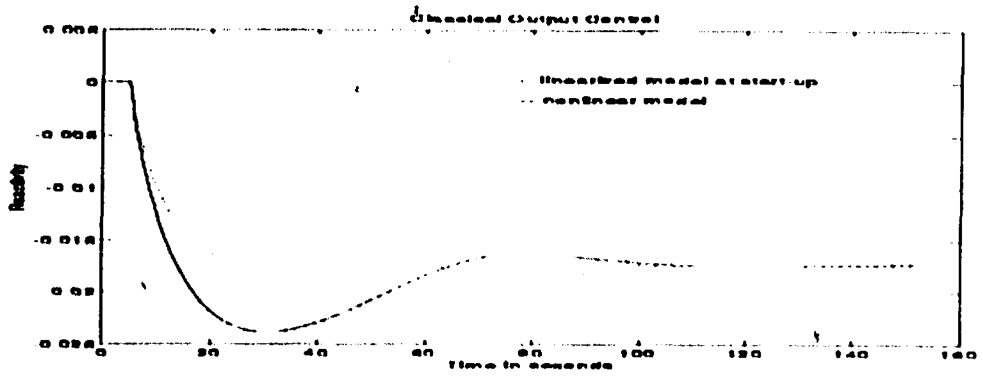


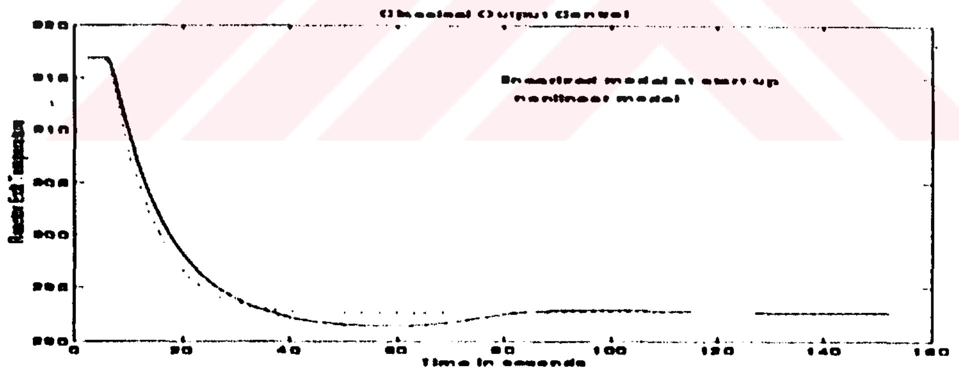
Fig 2.5.1.2. Dominant eigenvalues of linearized system(with temperature feedback) at operating points from 10% powçr to 150% power with equal incremental steps of %10 for loop gains of Gc=0.1, Gc=0.4, Gc=0.5, Gc=0.6.



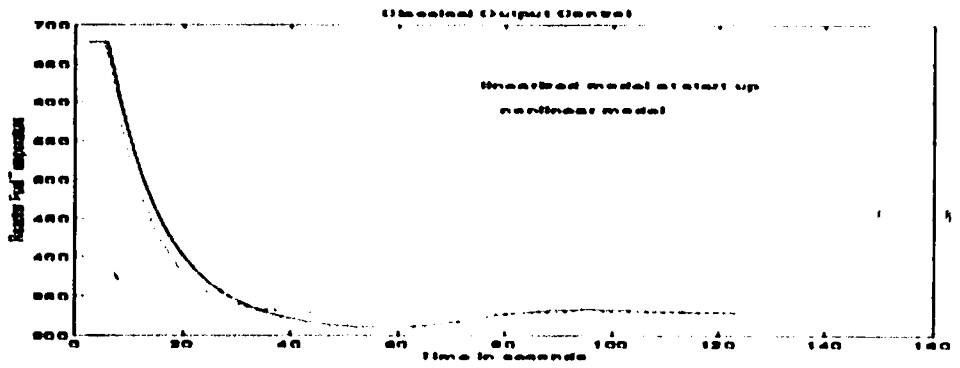
(a)


Classical Output Control
linearized model at start-up
nonlinear model
Reactivity
0.005
0
-0.005
-0.01
-0.015
-0.02
-0.025
0
20
40
60
80
100
120
140
160
Time in seconds

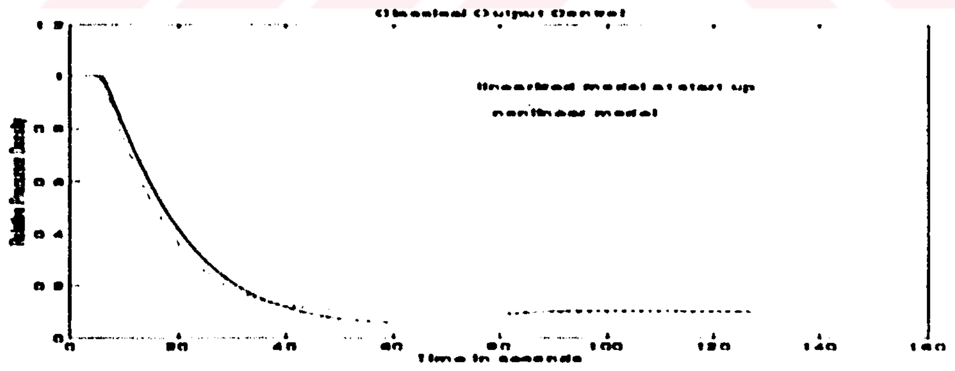
(b)


Classical Output Control
linearized model at start-up
nonlinear model
Reactor Exit Temperature
920
915
910
905
900
895
890
0
20
40
60
80
100
120
140
160
Time in seconds

(c)


Classical Output Control
linearized model at start up
nonlinear model
Reactor Fuel temperature
700
650
600
550
500
450
400
350
300
0
20
40
60
80
100
120
140
160
Time in seconds

(d)


Classical Output Control
linearized model at start up
nonlinear model
Relative Precursor Density
1.2
1
0.8
0.6
0.4
0.2
0
0
20
40
60
80
100
120
140
160
Time in seconds

(e)

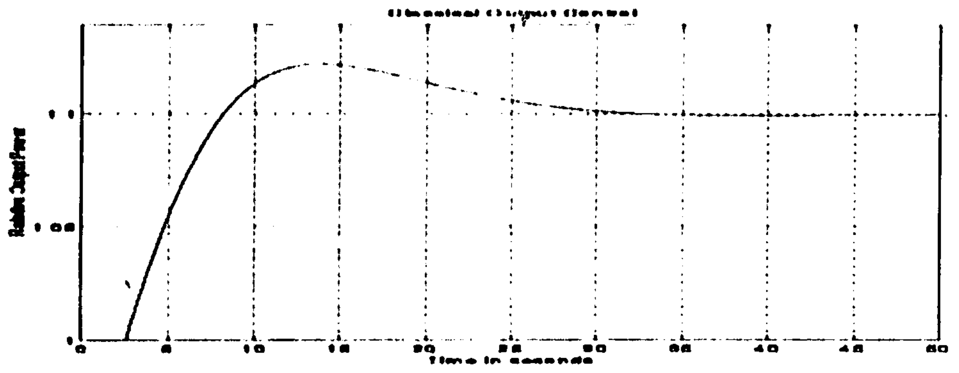
Fig 2.5.1.3 -(a),(b),(c),(d),(e) 90% drop in power is entered to classical control system , solid and dashed lines represent the simulation results of nonlinear and linearized model respectively.

Figures 2.5.1.1 and 2.5.1.2 show that temperature response of system is the major nonlinearity source. It is seen from Fig 2.5.1.1 that dominant eigenvalue of linearized system without temperature feedback is locatesd at the same path for increased gains $G_c$ and power levels. In the case of temperature feedback (in Fig 2.5.1.2), dominant eigenvalues are located at the different paths for changing gains $G_c$. These results can be turned out with a significant conclusion for the control of plants that response of linearized model without temperature feedback for different power levels can be compensated changing the value of gain $G_c$, but this is impossible for linearized models with temperature feedback.

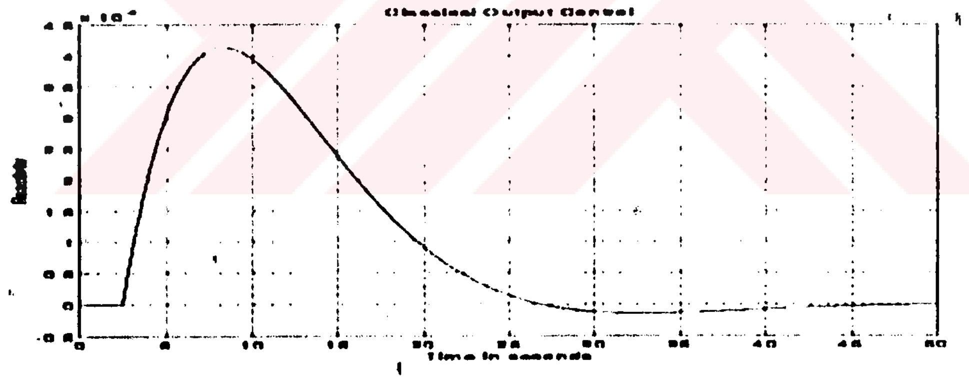
Results in Fig 2.5.1.3 show to what degree the linearized model variables match nonlinear model. Reactivity is the most sensitive variable to nonlinearity in linearized model as it is expected. The dynamical behaviors of linearized model are similar to nonlinear model for reactor relative power, temperatures and control rod speed. Hence, most significant deviation from nonlinear model is resulted from the reactivity.

### 2.5.2 Results for Classical Output Control

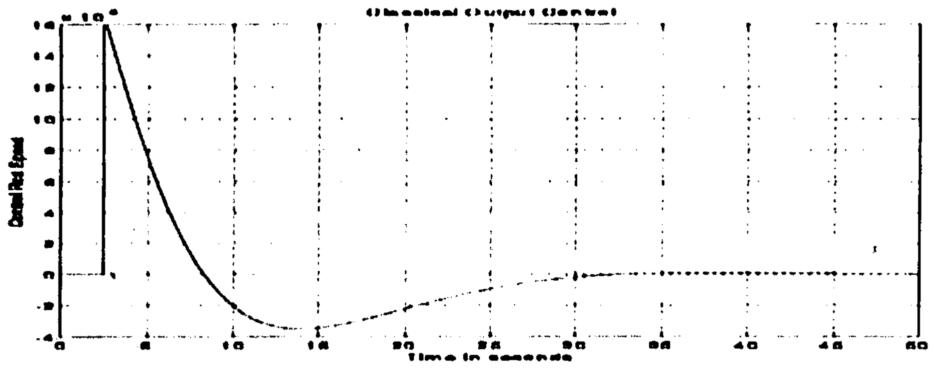
Results in Fig 2.5.2.1 illustrate the response of system when 10% power increase is entered to control stream without temperature feedback. Fig 5.2.1.2 illustrates results for temperature feedback . Both cases have been examined for nonlinear model of plant. Results of Fig 2.5.2.1 are obtained from Cont_1.m when Pd=1.1 ,$\alpha_f$=0, $\alpha_c$=0, $G_c$=0.1585 for Fig 2.5.2.2 when $G_c$=0.5 and $\alpha_f$, $\alpha_c$ are from table 2.1.


Classical Output Control
Relative Output Power
Time in seconds

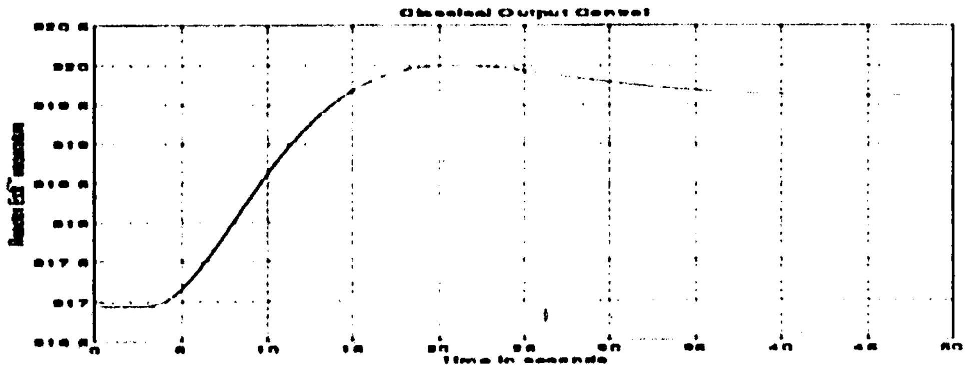
(a)


Classical Output Control
Reactivity
Time in seconds

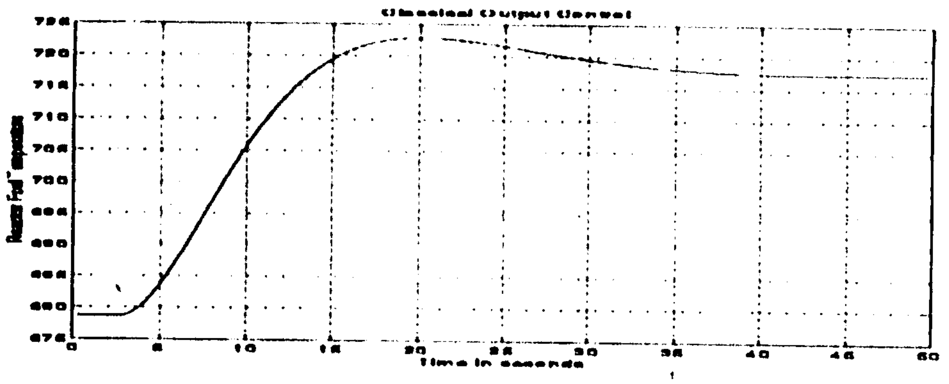
(b)


Classical Output Control
x 10⁻⁴
Control Rod Speed
Time in seconds

(c)

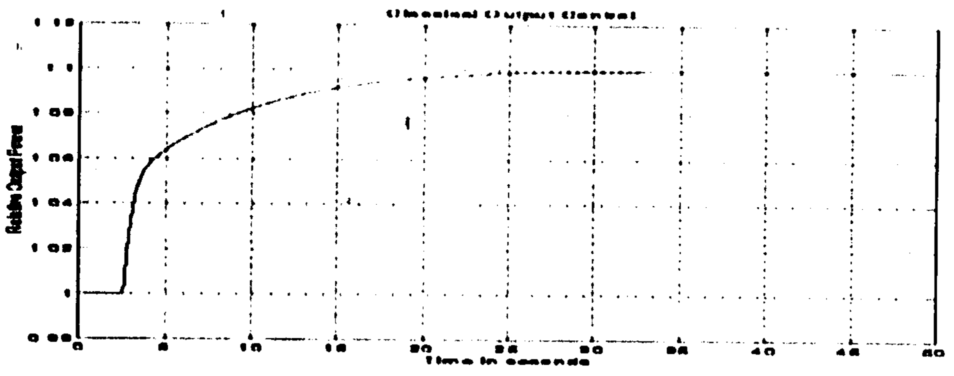

Classical Output Control
Reactor Exit Temperature
Time in seconds

(d)

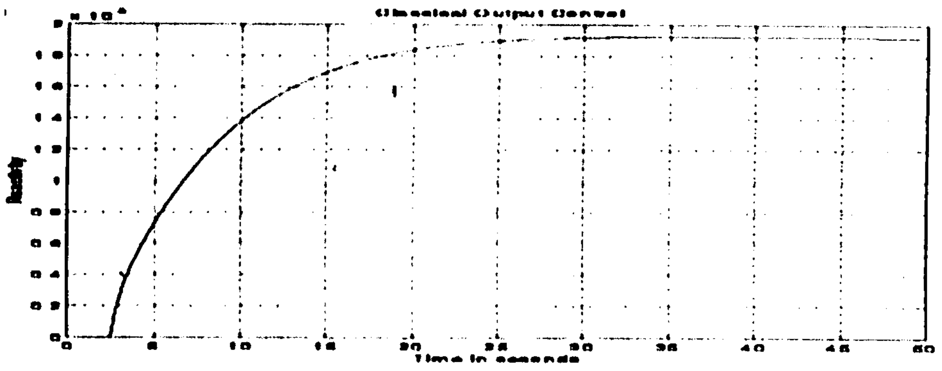


(e)

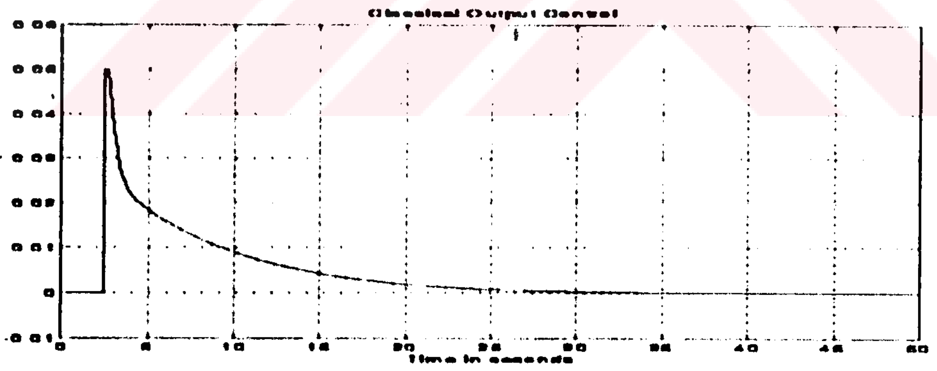
Fig 2.5.2.1 (a-b-c-d-e) Classical output control results without temperature feedback ($G_c$=0.1585) for a) relative output b)reactivity c) control rod speed d)reactor exit temperature e)reactor fuel temperature.



(a)



(b)



(c)

Fig 2.5.2.2 (a-b-c) Classical output control results with temperature feedback($P_d$=1.1 and $G_c$=0.5) for a)relative output power b) reactivity c) control rod speed.

### 2.5.3. Steady-State Error in COC.

Close loop transfer function with unitary feedback is

$$g_c(s) = \frac{G_c g(s)}{1 + G_c g(s)} = \frac{G_c n_r(s)}{z_r(s) + G_c n_r(s)} \qquad (2.5.3.1)$$

Using final value theorem;

$$y(t \to \infty) = \operatorname{Lim}_{s \to 0} s g_c(s) X(s) \qquad (2.5.3.2)$$

For step function input,

$$X(s) = P_d/s \qquad (2.5.3.3)$$

and steady-state output becomes

$$y(t \to \infty) = \operatorname{Lim}_{s \to 0} P_d g_c(s) = \operatorname{Lim}_{s \to 0} \frac{P_d G_c n_r(s)}{z_r(s) + G_c n_r(s)} = P_d \qquad (2.5.3.4)$$

$z_T = 0$ due to poles at zero for both with and without temperature feedback models (2.5.1), (2.5.2). Output at steady-state approaches demand power with zero error.

### 2.5.4. Stiffness in Nuclear Power Plant Model

Differential equations describing nuclear power plants are stiff equations that may result in numerical problems. To give a certain mathematical definition of stiffness is cumbersome undertaking. Stiff differential equations are required additional techniques to solve them. Numerical solvers for stiff ordinary differential equations are extensively studied. One of the most popular algorithms was proposed by Gear C.W.[51][53]. Gear's algorithm uses the backward difference formulas (BDF) [52] constructing a Newton backward-difference interpolating polynomial that is powerful tool for stiff problems. According to Gear, for the time-invariant case, a problem is stiff if the no eigenvalue has a real part which is at all large and positive and at least one eigenvalue has a real part which is large and negative. In stiff problem no solution component increases rapidly for large time span. However, there exist perturbations that cause rapidly changing solution components. Long time constants are dominant for a stiff problem (slowly varying solutions), but perturbations excite modes with small time constants (rapidly varying solutions) compared to time constants for the time span of interest. The large difference between time constants complicates the numerical solutions.

In numerical examples to find the solutions, stiff problems produce ill-conditioned matrices whose determinant is close to zero. Operations with ill-conditions matrices complicates the problem to find the correct solutions. For instance, taking inverse of a ill-conditioned matrix may result in incorrect solutions (see chapter six of [50]).

For the search of stiffness in nuclear power plant model, it is seen easily from mathematical model and simulation results that slowly varying output for critical reactor is rapidly damped for a perturbed reactivity input to reactor due to temperature feedback or control rod. The stiffness arising in nuclear power plant causes constraints in simulation of plants and controller design.

### 2.5.5 Conclusion

In COC, gain Gc has been calculated using root locus analysis to provide a good transient response. It is well understood that thermal response of system is a nonlinearity source (Fig 2.5.1.1 and Fig 2.5.1.2) and reactivity in the linearized model is the most deviated state variable from actual nonlinear model (Fig 2.5.1.3). It is also observed from Fig 2.5.2.2-a that controller designed for model with temperature feedback produced sluggish output response.

## 2.6. State Feedback Control with State Estimation (SFCSE)

In state feedback control, control signal is modified using all the state variables [4]. State variables amplified by gain K are fedback to system to modify the control signal. Main theorem that will be given in this section, guarantees the pole placement of plant mathematically while zeros appear with no change in the final transfer function. This and other useful concepts such as controllability and observability are also introduced in this section that these concepts will be frequently used in the remaining part of this work. Since we need state variables, and neutron density is the only measurable state variable, a state estimator is required to estimate the sates of plant.

### 2.6.1. State feedback Control with Pole Placement

*Definition 2.6.1.1: Control Canonical Form of State Space Equations*

Let's assume that A,B,C and D are arbitrarily chosen matrices with proper dimensions in the following state-space form.

$$\dot{x} = Ax + Bu \ , \qquad (2.4.1)$$

$$y = Cx + Du \qquad (2.4.2)$$

There is a linear transformation T between x and transformed x: $x_t$ that resultant transformed $A_t$, $B_t$ and $C_t$ are in the following forms respectively.

$$A_t = \begin{bmatrix} 0 & 1 & . & 0 & 0 \\ 0 & 0 & . & 0 & 0 \\ 0 & 0 & . & 1 & 0 \\ 0 & 0 & . & 0 & 1 \\ -a_0 & -a_1 & . & -a_{n-2} & -a_{n-1} \end{bmatrix}_{n\times n} , \quad _t = \begin{bmatrix} 0 \\ 0 \\ . \\ 0 \\ 1 \end{bmatrix}_{n\times 1} , \qquad (2.6.1.1)$$

$$C = \begin{bmatrix} b_0 & b_1 & , & b_{n-2} & b_{n-1} \end{bmatrix}_{1\times n}$$

$A_t$, $B_t$ and $C_t$ represent the control canonical form and transformation T exist if and only if A and B are controllable.

*Theorem 2.6.1.1*

Transfer function of the system may be written directly from the control canonical form as

$$G(s) = \frac{b_{n-1}s^{n-1} + b_{n-2}s^{n-2} + \cdots + b_1 s + b_0}{s^n + a_{n-1}s^{n-1} + a_{n-2}s^{n-2} + \cdots + a_1 s + a_0} \quad \text{where n is the size of A.}$$

*Theorem 6.2.1.2 Cayley-Hamilton Theorem*

Characteristic function of a system matrix A is

$|\lambda I - A| = \lambda^n + a_{n-1}\lambda^{n-1} + \cdots + a_1\lambda + a_0$ . where coefficients $a_i$ are from $A_t$ in control canonical form.

*Definition 6.2.1.2 Similarity Transformation*

let's assume that A, B and C are arbitrarily given in (2.4.1)

$$\dot{x} = Ax + Bu \ , \qquad (2.4.1)$$

$$y = Cx$$

Then

$$P = \begin{bmatrix} B & AB & A^2B \cdots A^{n-1}_{n\times n} B_{n\times 1} \end{bmatrix}_{n\times n} \qquad (2.6.1.2)$$

let's assume that $P^{-1}$ exists

$$P^{-1} = \begin{bmatrix} \cdot \\ \cdot \\ \cdot \\ \cdot \\ q_{1xn} \end{bmatrix}_{nxn}$$ , where $q_{1xn}$ is the last row of $P^{-1}$. (2.6.1.3)

define

$$T^{-1} = \begin{bmatrix} q_{1xn} \\ q_{1xn}A_{nxn} \\ \cdot \\ \cdot \\ q_{1xn}A_{nxn}^{n-1} \end{bmatrix}_{nxn} \qquad (2.6.1.4)$$

$x_t = T^{-1}x$ transforms the system A,B and C into system $A_t$,$B_t$ and $C_t$ that

$A_t = T^{-1}A\,T$, $B_t = T^{-1}B$, $C_t = C\,T$ in control canonical form.

Augmented state space form of (2.4.1) and (2.4.2) with state feedback loop gain K;

$\dot{x} = (A - BK)x + Bu$ , (2.6.1.5)

$y = C\,x$

Original equation appears with no change except (A-BK) takes place of A as a new system matrix.

*Notes on Similarity Transformation for State Feedback*

State feedback does only change places of poles, but does not change the zeros.

This may be proved using similarity transformation and linear system theory. Similarity transformation does not change the transfer function.

*Proof:*

Transfer function of system (2.4.1) and (2.4.2) is found as

$$G(s) = \frac{Y(s)}{X(s)} = C(sI - A)^{-1}B \qquad (2.6.1.6)$$

using Laplace transformation and ignoring coupling matrix D.

Transfer function of transformed system is

$$G_t(s) = \frac{Y_t(s)}{X_t(s)} = C_t(sI - A_t)^{-1}B_t \qquad (2.6.1.7)$$

$$G(s) = C_t T^{-1}(sI - TA_t T^{-1})^{-1} TB_t$$

$$G(s) = C_t T^{-1}(T(sI - A_t)T^{-1})^{-1} TB_t = C_t T^{-1} T(sI - A_t)^{-1} T^{-1} TB_t$$

$$G(s) = C_t (sI - A_t)^{-1} B_t = G_t(s)$$

*Definition 2.6.1.3: Controllability:*

Linear time-invariant(LTI) system

$$\dot{x} = Ax + Bu \,, \qquad (2.4.1)$$
$$y = Cx$$

is controllable if any control input transfer the initial state x(0) at t=0 into zero state $x(t_0)=0$ in finite time $t_0$. There are several ways to define controllability; if there exist control canonical form of (2.4.1) system is controllable. To satisfy this condition rank of P defined in similarity transformation must be equal to it's size or inverse of P must exist. In this case, (A,B) is said to be controllable.

$$P = \begin{bmatrix} B & AB & A^2B \cdots A^{n-1}_{nxn}B_{nx1} \end{bmatrix}_{nxn} \,.$$

These theorems guarantee the pole placement of system that desired poles may be placed adjusting feedback loop gain K. Even an unstable system may be stabilized choosing proper poles at left hand side. Similarity transformation is powerful tool to calculate K properly in linear system theory. However J.E. Ackermann formulated state feedback K in following formula ;

$$K = [0\ 0\ \ldots.\ 0\ 1][B\ AB\ \ldots\ldots\ A^{n-2}B\ A^{n-1}B]^{-1}\alpha_c(A) \qquad (2.6.1.8)$$

is known as Ackermann's formula.

where $\alpha_c(A)$ is matrix polynomial with the coefficient of desired characteristic equation (desired poles to be placed);

$$\alpha_c(A) = A^n + \alpha_{n-1}A^{n-1} + \ldots\ldots + \alpha_1 A + \alpha_0 I \qquad (2.6.1.9)$$

**2.6.2. State Estimation**

With this section, we will introduce the deterministic linear observer design (Luenberger's Observer).

*Definition: Observability*

Linear time-invariant(LTI) system

$$\dot{x} = Ax + Bu \,, \qquad (2.4.1)$$
$$y = Cx$$

is said to be observable if the initial conditions x(t=0) can be determined from the output function y(t) in finite time.

It may also be shown that the observability of system depends on the availability of inverse of following matrix;

$$R = \begin{bmatrix} C_{nx1} \\ C_{nx1}A_{nxn} \\ \cdot \\ C_{nx1}A_{nxn} \\ C_{nx1}A_{nxn} \end{bmatrix}_{nxn} , \qquad (2.6.2.1)$$

In another definition, (A,C) is said to be observable if rank(R)=n .
Note that (A,C) is observable if $(A^1,C^1)$ is controllable.

Similarly, necessary mathematical derivation for obsevability may be proved using linear matrix algebra and defining observer canonical form. But we skip the proof and present the Ackermann's formula to determine linear observer gain H;

$$H = \alpha_e(A) \begin{bmatrix} C \\ CA \\ \cdot \\ CA^{n-2} \\ CA^{n-1} \end{bmatrix}_{nxn}^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \qquad (2.6.2.2)$$

where $\alpha_e(A)$ is the matrix polynomial with proper coefficients that define the desired pole placement for observer.

## 2.6.3. The Design of State Feedback Control with State Estimation (SFCSE)

The Fig 2.6.3.1 illustrates the whole system for state feedback control with state estimation. The feedback loop existing in classical control has been saved in state feedback control. This enables a reconfigurable controller design from classical output control to state feedback switching to which controller type is preferred. A is replaced with the A-$B_cC$ where $B_c$=Gc B and Gc substitutes loop gain in COC and appears as a constant value of 0.5 throughout SFCSE. $A_c$ represents the new system matrix with classical loop:

$$A_c=A-B_cC \qquad (2.6.3.1)$$

It was noted in COC that improving dynamic response of system will not be possible and it is determined as a main restriction in already existing design methodology.

Theories defined in this section enable us to replace the plant's poles to provide the desired response. -0.2381 and -1.373 are chosen to cancel the zeros in transfer function of linearized model with temperature feedback (2.5.2). $-0.122\pm0.125$ j is the dominant pole for a damping ratio of 0.7 while -64.4 is chosen as a final pole with no change. Gain K is determined using Ackermann's formula defined in (2.6.1.8) with the following characteristic equation:.

$\alpha(A_c)= A_c^5+66.2551\, A_c^4+120.218\, A_c^3+48.4629\, A_c^2+8.31238\, A_c+0.642308\, I_{5x5}$ (2.6.3.2)

and resultant gain K is

$K=[-0.9895 \;\; -0.6267 \;\; 0.0074 \;\; -0.0031 \;\; -96.1626]$ (2.6.3.3)

Poles of estimator are chosen taking all the poles with no change except dominant pole to provide a faster response than the response of state feedback control. Characteristic equation with poles at $-025\pm$ j 0.25, -0.2381, -1.373 , -64.40

$\alpha_e(A_c)= A_c^5 +66.5111\, A_c^4+137.212\, A_c^3+81.3453\, A_c^2+23.5367\, A_c+2.63163\, I_{5x5}$ (2.6.3.4)

and resultant gain H for Luenberger's observer.

$$H=[-0.2248; \quad -0.4933; \quad 93.1370; \quad 0.7743; \quad 0.0014] \qquad (2.6.3.4)$$

Non-dynamical precompensator v is determined using final value theorem to track the demand power. Before calculating v, we will present fundamental model for general state feedback with state estimation to explain and ensure the dynamic behavior of system

For estimator

$\dot{\hat{x}} = (A_c - HC)\hat{x} + Bu + Hy$ (2.6.3.5)

$y = C\hat{x}$ where $\hat{x}$ is estimated state vector.

and state feedback

$\dot{x} = A_c x + Bu$ , (2.6.3.6 )

$y = C x$

Combining state feedback with state estimation in Fig 2.6.3.1 new system dynamics becomes;

$\dot{x} = A_c x - BK\hat{x} + Bv\delta P_d$ (2.6.3.7)

$$\dot{\hat{x}} = (A_c - HC - BK)\hat{x} + HCx + Bv\delta P_d \qquad (2.6.3.8)$$

Equations (2.6.3.7) and (2.6.3.8) may be rewritten in matrix form;

$$\begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix} = \begin{bmatrix} A_{nxn} & -B_{nx1}K_{nx1} \\ H_{1xn}C_{nx1} & (A_{nxn} - H_{1xn}C_{nx1} - B_{nx1}K_{nx1}) \end{bmatrix} \begin{bmatrix} x \\ \hat{x} \end{bmatrix} + \begin{bmatrix} B_{nx1} \\ B_{nx1} \end{bmatrix} v\delta P_d \qquad (2.6.3.9)$$

Error vector $x_e = x - \hat{x}$ is defined in matrix form

$$\begin{bmatrix} x \\ x_e \end{bmatrix} = \begin{bmatrix} I_{nxn} & 0 \\ I_{nxn} & -I_{nxn} \end{bmatrix} \begin{bmatrix} x \\ \hat{x} \end{bmatrix} \qquad (2.6.3.10)$$

Also using derivatives of equation (2.6.3.10)

$$\begin{bmatrix} \dot{x} \\ \dot{x}_e \end{bmatrix} = \begin{bmatrix} I_{nxn} & 0 \\ I_{nxn} & -I_{nxn} \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix} \qquad (2.6.3.11)$$

and following operations

$$\begin{bmatrix} x \\ \hat{x} \end{bmatrix} = \begin{bmatrix} I_{nxn} & 0 \\ I_{nxn} & -I_{nxn} \end{bmatrix}^{-1} \begin{bmatrix} x \\ x_e \end{bmatrix} = \begin{bmatrix} I_{nxn} & 0 \\ I_{nxn} & -I_{nxn} \end{bmatrix} \begin{bmatrix} x \\ x_e \end{bmatrix} \qquad (2.6.3.12)$$

$$\begin{bmatrix} I_{nxn} & 0 \\ I_{nxn} & -I_{nxn} \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{x}_e \end{bmatrix} = \begin{bmatrix} A_{nxn} & -B_{nx1}K_{nx1} \\ H_{1xn}C_{nx1} & (A_{nxn} - H_{1xn}C_{nx1} - B_{nx1}K_{nx1}) \end{bmatrix} \begin{bmatrix} I_{nxn} & 0 \\ I_{nxn} & -I_{nxn} \end{bmatrix} \begin{bmatrix} x \\ x_e \end{bmatrix} + \begin{bmatrix} B_{nx1} \\ B_{nx1} \end{bmatrix} v\delta P_d$$

(2.6.3.13)

resultant system dynamics is governed by

$$\begin{bmatrix} \dot{x} \\ \dot{x}_e \end{bmatrix} = \begin{bmatrix} A_{nxn} - B_{nx1}K_{nx1} & B_{nx1}K_{nx1} \\ 0 & (A_{nxn} - H_{1xn}C_{nx1}) \end{bmatrix} \begin{bmatrix} x \\ x_e \end{bmatrix} + \begin{bmatrix} B_{nx1} \\ 0 \end{bmatrix} v\delta P_d \qquad (2.6.3.14)$$

The governing equation (2.6.3.14) shows that error dynamics is an autonomous system with no input. Error vector $x_e$ vanishes exponentially for the eigenvalues located at the left hand side properly choosing observer gain H in system matrix A-HC. This provides and proves the stability of estimator's error asymptotically. When the error of estimator vanishes at steady state, system is governed by system matrix A-BK and input change in $P_d$ .To track the demand power at output, v is determined at steady-state conditions when $x_e$=0 than close loop transfer function is

$$g(s) = \frac{\delta P}{\delta P_d} = \frac{50(s+0.125)}{(s+0.122 \pm j0.125)(s+64.40)} v$$

To satisfy $\delta P = \delta P_d$ at steady-state

$$\delta P(t \to \infty) = \mathrm{Lim}_{s \to 0} sg(s)\delta P_d(s) = \frac{50(s+0.125)}{(s+0.122 \pm j0.125)(s+64.40)} v\delta P_d = 3.1183 v\delta P_d$$

v = 1/ 3.1183 to track the demand power.

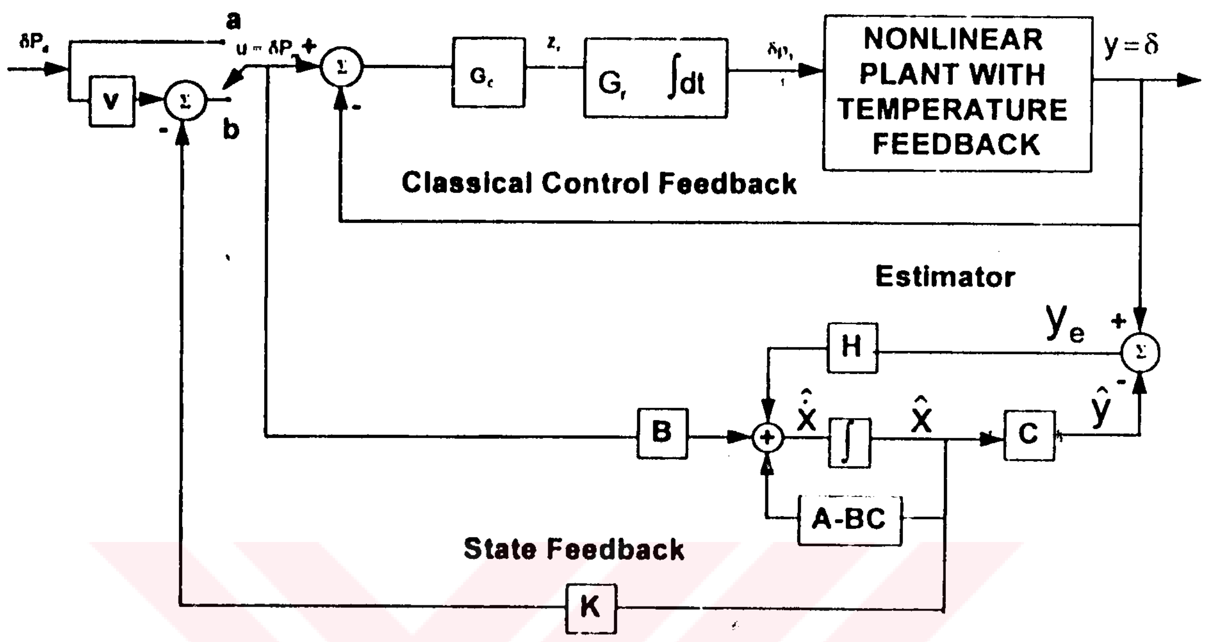


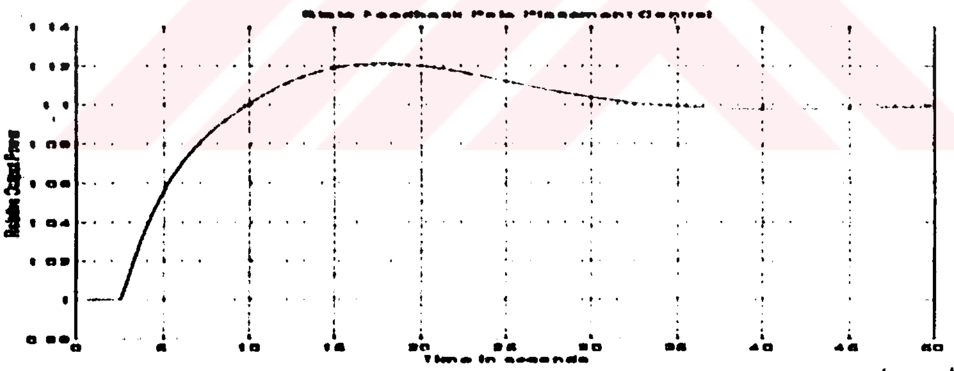
Fig 2.6.3.1 State feedback control with state estimation, reconfigurable control from COC to SFCSE. When switch a is turned on control system is SFCSE, otherwise COC is employed without any change in design parameters.

## 2.6.4 Simulation of SFCSE for Nuclear Power Plant

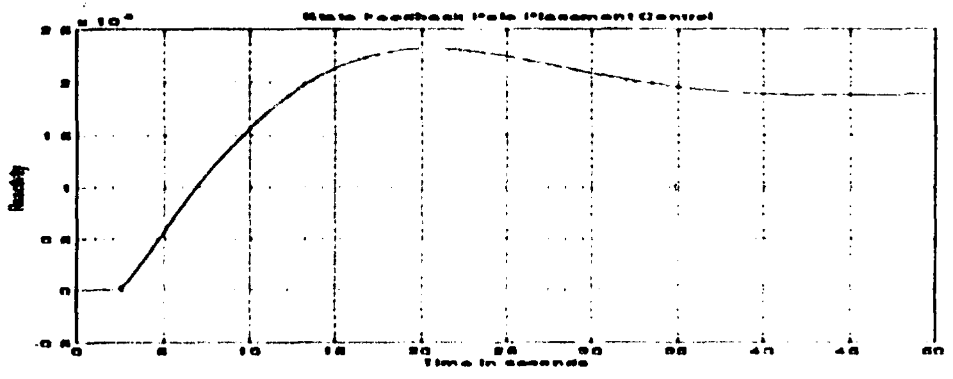
Matlab code csfsc_1.m has been run to simulate the SFCSE for PWR illustrated in fig 2.6.3.1. Code csfsc_1.m used state feedback K and observer gain H that are determined in equations (2.6.3.3) and (2.6.3.4) respectively. The simulation results are illustrated in fig 2.6.3.2 for $G_c$=0.5 and constants from table 2.1.

Table 2.1 Values of Constants Used in Chapter Two

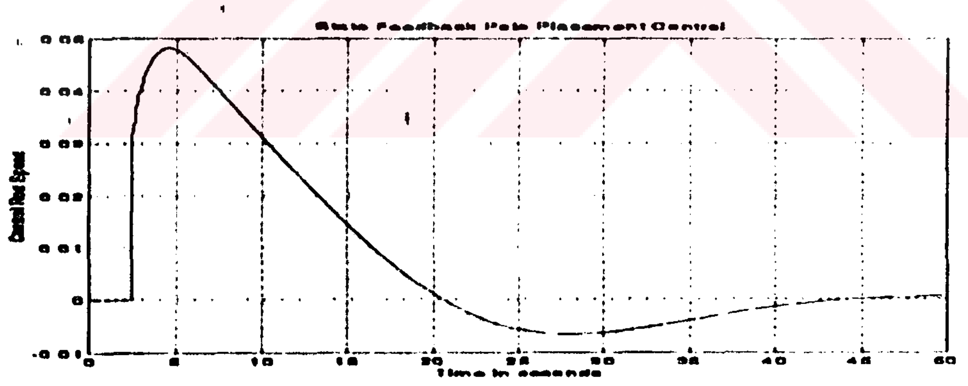
| | | | |
|---|---|---|---|
| $\beta$ | 0.0065 | $\lambda$ | 0.125 $s^{-1}$ |
| $\Lambda$ | 0.0001 s | $f_f$ | 0.98 |
| $G_r$ | 0.01 total rod reactivity | $T_e$ | 290 $C^0$ |
| $P_{oa}$ | 2500 MW | $\mu_c$ | 70.5 MW.s/$C^0$ |
| $\mu_f$ | 26.3 MW.s/$C^0$ | M | 92.8 MW/$C^0$ |
| $\Omega$ | 6.53 MW.s/$C^0$ | $\alpha_c$ | +0.00001 reactivity/$C^0$ |
| $\alpha_f$ | -0.00005 reactivity/$C^0$ | | |



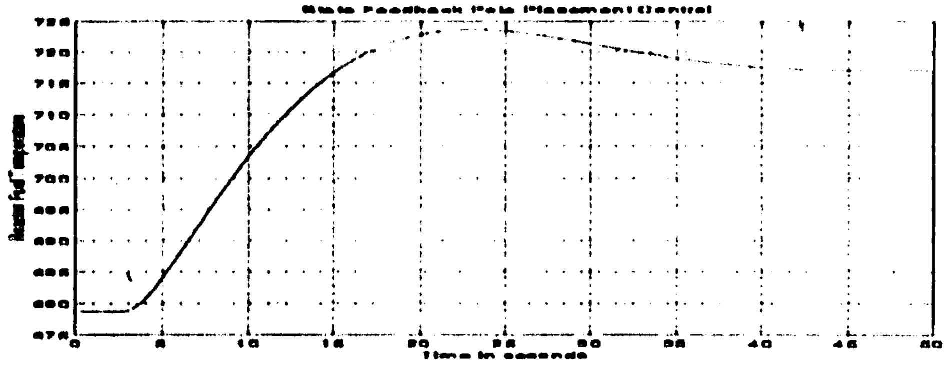
(a)


State Feedback Pole Placement Control
x 10^-3
Reactivity
2.5
2
1.5
1
0.5
0
-0.5
0
5
10
15
20
25
30
35
40
45
50
Time in seconds

(b)

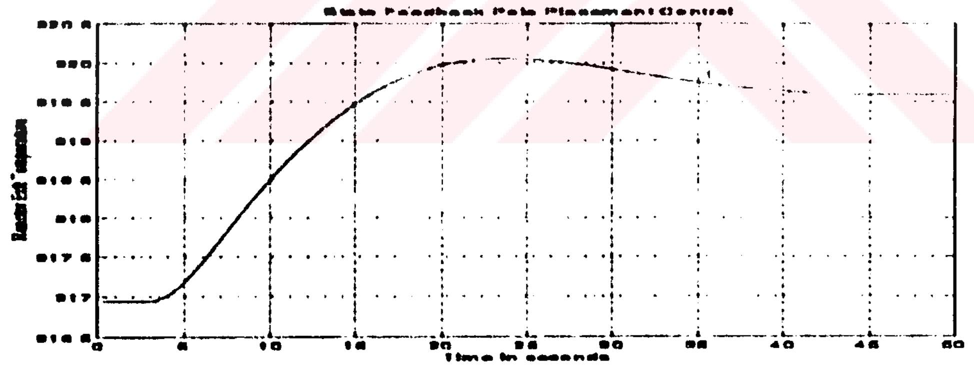

State Feedback Pole Placement Control
Control Rod Speed
0.05
0.04
0.03
0.02
0.01
0
-0.01
0
5
10
15
20
25
30
35
40
45
50
Time in seconds

(c)



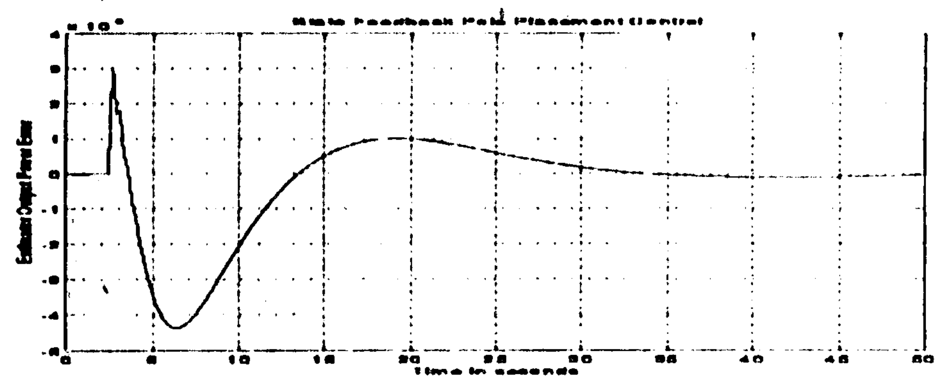
State Feedback Pole Placement Control
725
720
715
710
705
700
695
690
685
680
675
0
5
10
15
20
25
30
35
40
45
50
Time in seconds

(d)

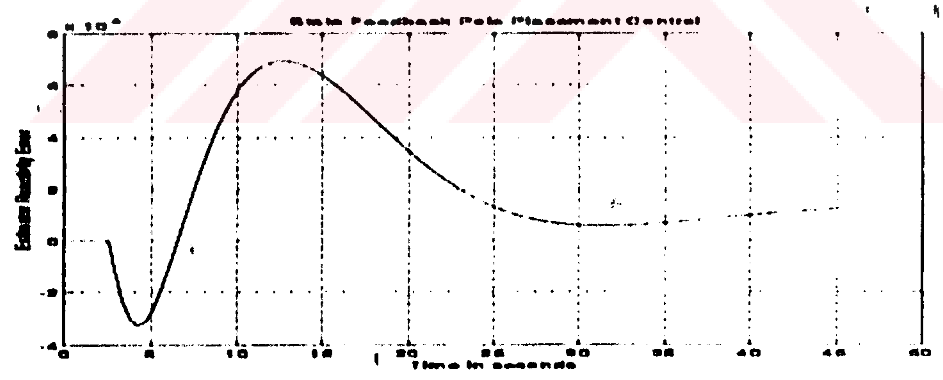


State Feedback Pole Placement Control
820.5
820
819.5
819
818.5
818
817.5
817
816.5
0
5
10
15
20
25
30
35
40
45
50
Time in seconds

(e)



(f)



(g)

Fig 2.6.3.2 Simulation results for state feedback control with state estimation a) relative output power (%10 increase in demand power is entered to system) b)reactivity c)control rod speed d)reactor fuel temperature e) estimator output power error d) estimator reactivity error.

## 2.6.5 Conclusion

SFCSE has been designed using pole placement techniques and examined for the nonlinear model of plant. As it is expected from the state feedback control theory, the desired response has been obtained by placing the poles for the model with temperature feedback. From fig 2.6.3.2-a, the same output response is obtained for model with temperature response, whereas it was sluggish in COC (fig 2.5.2.2-a). State estimator (Luinberger's observer) has found the states with different steady-state errors. While relative output power is estimated with zero steady-state error, the estimated value of reactivity does not shrink to actual reactivity value in nonlinear model. This is mainly due to inherent nonlinearity of reactivity. It is concluded from section 2.5.1 that most significant error between nonlinear and linearized model occurs for reactivity. Since we designed a linear observer using linearized model at start-up, it can not estimate the reactivity with a zero steady-state error for different operating power levels. Hence far, the tools to change the poles (so, the response of plant) and design linear observer have been presented, but no method has been suggested to determine the places of poles for an optimal response.

# CHAPTER THREE

# OPTIMAL CONTROL

In this chapter we will focus on optimal control [9][11] and it's applications to nuclear plants [1][6][7][8]. Linear quadratic optimal control would be a more descriptive title for this chapter. Despite it is possible to develop nonlinear optimal control laws, linear or linearized models will be assumed in design stage and quadratic cost functions will be employed due to their mathematical simplicity in derivation of optimal control laws.

## 3.1. Linear Regulator Problem

Some important definitions which we shall introduce next [13] [12] will play a significant role in problem statement. All of our definitions are assumed for the following linear time-invariant deterministic dynamical system:

$$\dot{x}(t) = Ax(t) + Bu(t) \qquad (3.1.1)$$
$$y(t)=Cx(t)+Du(t)$$

with the given initial condition x(0) and state $x(t) \in \Re^n$.

*Definition 3.1.1*

A state $x \in \Re^n$ is said to be A-stable if

$\lim_{t\to\infty} e^{At}=0$

A necessary and sufficient condition is that all the eigenvalues of A must be located at the left hand side or equivalently real parts of eigenvalues must be less than zero.

*Definition 3.1.2*

A state $x \in \Re^n$ is said to be unobservable if

$C e^{At} x=0 \quad t\geq 0$

Note that if there is no unobservable state or observable space is the whole $\Re^n$ (A,C) is said to be observable from definition(2.6.2.1).

*Definition 3.1.3*

(A,C) is said to be detectable if all unobservable states of (3.1.1) are A-stable from definition (3.1.1).

*Definition 3.1.4*

(A,B) is said to stabilizable if there exist a linear state feedback control input u=Kx that makes (A-BK)-stable. (A,B) is stabilizable if and only if $(A^T, B^T)$ is detectable.

*Definition (3.1.5): Definite Quadratic Forms*

If $x^TAx>0$ for all real x except x=0, $x^TAx$ is said to be positive definite.

If $x^TAx\geq 0$ for all real x except x=0, $x^TAx$ is said to be positive semidefinite.

If $x^TAx<0$ for all real x except x=0, $x^TAx$ is said to be negative definite.

If $x^TAx\leq 0$ for all real x except x=0, $x^TAx$ is said to be negative semidefinite.

A quadratic form $x^TAx$ is said to be indefinite if the form is negative for some points x and positive for others [83].

A symmetric matrix $A_{nxn}$ is said to be positive definite, positive semidefinite and negative definite etc., if the relative quadratic term $x^TAx$ is positive definite, positivo semidefinite and negative definite.

$x^TAx$ is positive (negative) definite if and only if every eigenvalue of A is positive (negative).

$x^TAx$ is positive (negative) semidefinite if and only if all eigenvalues of A are non-negative (nonpositive) and at least one of the eigenvalues vanishes.

$x^TAx$ is indefinite if and only if some eigenvalues of A are positive and others are negative.

*Regulator problem:*

For time-varying model

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) \qquad (3.1.2)$$

The following quadratic performance index is defined to be minimized

$$J(x(t_o),u(.),t_o) = \int_{t_o}^{T}(u^TRu) + x^TQx)dt + x^T(T)Gx(T) \qquad (3.1.3)$$

A(t), B(t), Q(t) and R(t) are assumed to be continuos and Q(t) and R(t) are symmetric nonnegative and positive definite. Also G is a nonnegative definite symmetric matrix. Finding an optimal $u^*(t), t\in[t_0,T]$ that minimizes performance index $J((t_0),u(.),t_0)$ defined in (3.1.3) and the related optimum performance index $J^*x(t_0),u(.),t_0)$ is the main subject in regulator problem. When T is finite regulator problem becomes a finite time (finite horizon) problem. If T is infinite, problem designated as an infinite time or infinite horizon problem [9].

To solve this problem (indeed problems), several optimization theorems such as Minimum Principle of Pontryagin and Euler-Langrange Equations may be used as inherent optimization problem tools. The Hamilton-Jacobi equation is another tool introduced in the next section to solve the optimal control problems.

## 3.2 The Hamilton-Jacobi Equation

For the system in general nonlinear differential state-space form [9]

$\dot{x} = f(x,u,t)$ with given initial condition $x(t_0)$ (3.2.1)

to find the optimal $u^*(t)$, $t\in[t_0,T]$, which minimizes the performance index (cost function)

$$J(x(t_0),u(.),t_0) = \int_{t_0}^{T} l(x(\tau),u(\tau),\tau)d\tau + m(x(T)). \qquad (3.2.2)$$

Starred J represents the minimum(optimal) cost function according to

$$J^*(x(t),t) = \min_{u[t,T]} J(x(t),u(.),t) \qquad (3.2.3)$$

$$J^*(x(t),t) = \min_{u[t,T]}\left[\int_{t}^{T} l(x(\tau),u(\tau),\tau)d\tau + m(x(T))\right] \qquad (3.2.4)$$

from definition t is an arbitrary time in the range $[t_0,T]$

$$J^*(x(t_0),t_0) = \min_{u[t_0,t]}\left\{\min_{u[t,T]}\left[\int_{t_0}^{t} l(x(\tau),u(\tau),\tau)d\tau + \int_{t}^{T} l(x(\tau),u(\tau),\tau)d\tau + m(x(T))\right]\right\}$$

(3.2.5)

The first term in summation is independent of $u[t,T]$,therefore inner min becomes

$$J^*(x(t_o),t_o) = \min_{u[t_o,t]}\left\{ \int_{t_o}^{t} l(x(\tau),u(\tau),\tau)d\tau \quad + \min_{u[t,T]}\left[ \int_{t}^{T} l(x(\tau),u(\tau),\tau)d\tau \quad + m(x(T))\right]\right\}$$

(3.2.6)

The second term in inner summation is equal to optimal performance index;

$$J^*(x(t_o),t_o) = \min_{u[t_o,t]}\left\{ \int_{t_o}^{t} l(x(\tau),u(\tau),\tau)d\tau \quad + J^*(x(t),t)\right\} \tag{3.2.7}$$

which is an expression of Principle of Optimality used frequently to find optimal control laws.

Let's assume that $t = t_o+\Delta t$ where $\Delta t$ is infinitesimal, in this case equation (3.2.7) becomes

$$J^*(x(t_o),t_o) = \min_{u[t_o,t_o+\Delta t]}\left\{ \int_{t_o}^{t_o+\Delta t} l(x(\tau),u(\tau),\tau)d\tau \quad + J^*(x(t_o+\Delta t),t_o+\Delta t)\right\}$$

(3.2.8)

using Taylor's expansion for the second term in summation;

$$f(t_o+\Delta t) = f(t_o) + \left.\frac{\partial f(t)}{\partial t}\right|_{t=t_o} \Delta t + \text{OtherTerms}$$

$$J^*(x(t_o+\Delta t),t_o+\Delta t) = J^*(x(t_o),t_o) + \left[\frac{\partial J^*}{\partial x}(x(t_o),t_o)\right]^T \frac{\partial x(t_o)}{\partial t}\Delta t + \frac{\partial J^*}{\partial t}(x(t_o),t_o)\Delta t + 0\cdot\Delta t^2$$

(3.2.9)

Replacing (3.2.9) into (3.2.8) following equation is obtained as

$$J^*(x(t_o),t_o) = \min_{u[t_o,t_o+\Delta t]}\Big\{\Delta t\cdot l(x(t_o+\Delta t, u(t_o+\Delta t),t_o+\Delta t) + J^*(x(t_o),t_o)$$
$$+\left[\frac{\partial J^*}{\partial x}(x(t_o),t_o)\right]^T \frac{\partial x(t_o)}{\partial t}\Delta t + \frac{\partial J^*}{\partial t}(x(t_o),t_o)\Delta t + 0\cdot\Delta t^2\Big\}$$

(3.2.10)

After necessary operations Hamilton-Jacobi equation is obtained by

$$\frac{\partial J^*}{\partial t} = -\min_{u(t)}\left\{ l(x(t),u(t),t) + \left[\frac{\partial J^*}{\partial x}\right]^T f(x(t),u(t),t)\right\}$$

(3.2.11)

## 3.3. Solution of Optimal Control Problem

We give the priority to discrete-time systems for derivation of optimal control solutions because of their explanatory facilities in mathematical comprehension of optimal control theory based on principle of optimality. In the limit case while sampling in discrete time goes to infinity, similar to the derivation of Hamilton-Jacobi equation from principle of optimality in previous section, optimal control results are obtained for continuos-time systems. It may be shown that Hamilton-Jacobi equation may be used directly to derive the optimal control laws for continuos-time case.

### 3.3.1 Discrete Time Systems

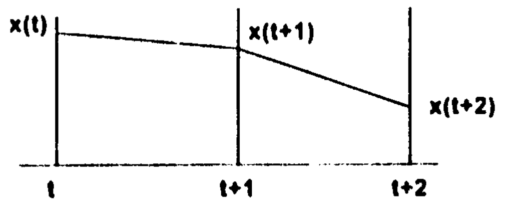
Discrete-time state-space equation

$$x(t+1) = A(t)\,x(t) + B(t)\,u(t) \qquad (3.3.1.1)$$

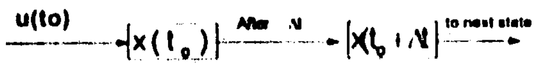
with initial condition $x(t_0)$ given.

In this case, performance index defined in Eq.(3.1.3) becomes

$$J(x(t_0),u(.),t_0) = \sum_{t=t_0+1}^{T}(u^T(t-1)R(t)u(t-1)) + x^T(t)Q(t)x(t)) \qquad (3.3.1.2)$$



Optimal Control Problem



or equivalent of Eq.(3.3.1.2) is

$$J(x(t_o),u(.),t_o) = \sum_{t=t_o}^{T-1}\left(u^T(t)R(t+1)u(t)) + x^T(t)Q(t)x(t)\right) - x^T(t_o)Q(t_o)x(t_o) + x^T(T)Q(T)x(T) \quad (3.3.1.3)$$

With these evaluations, we will interpret the optimal control philosophy under the light of principle of optimality formulated in Eq.(3.2.7) for discrete case, remembering that continuos time may be obtained from discrete time in the limit of $\Delta t \to 0$.

We apply $u(t_o)$ as a control input, at time $t_o$ states of the system is $x(t_o)$. After $\Delta t$, $t=t_o+\Delta t$ as a response to $u(t_o)$ states of system becomes $x(t_o+\Delta t)$. Hence, the optimal control problem is that how can we find an optimal input sequence $u(t_o)$, $u(t_o+\Delta t)$, $u(t_o+2\Delta t)$,.... $u(t_o+k\Delta t)$ that minimizes the performance index below ?

For every sample, cost is

$$K(t_o+\Delta t) = \left[x^T(t_o+\Delta t)Qx(t_o+\Delta t) + u^T(t_o)R(t_o+\Delta t)u(t_o)\right]\cdot \Delta t \quad (3.3.1.4)$$

(Performance index at $t_o+\Delta t$)

Note that we applied input $u(.)$ at $t_o$ and waited for the response and measured the states $x(t_o+\Delta t)$. With this available data, we calculated performance index defined in Eq.(3.3.1.4).

The whole cost function is a summation of performance indices determined for every sample.

$$J = \sum_{k=1}^{n} K(t_o+k\Delta t)\cdot \Delta t \qquad T = t_o + n\cdot\Delta t \qquad (3.3.1.5)$$

In the continuos case, $\Delta t \to 0$

$$J(x(t_o),u(.),t_o) = \int_{t_o}^{T}(u^TRu) + x^TQx)dt + x^T(T)Gx(T) \qquad (3.1.3)$$

let's take $\Delta t=1$ as unity in this case

$$J = \sum_{k=1}^{n} K(t_o+k) \qquad T = t_o + n \qquad (3.3.1.6)$$

Also it is a priori knowledge that optimum performance index is in the following form;

$J^*(x(t_o),t_o)= x^T(t_o)\,P(t_o)\,x(t_o)$ where $P(t)$ is symmetric matrix. (3.3.1.7)

$J^*(x(t),t)=\min_{u(t)}[\,K(t_o+1) + J^*(x(t+1),t+1)]=x(t)^TP(t)x(t)$ (3.3.1.8)

$x(t+1) = A(t)\, x(t) + B(t)\, u(t)$, $J^*(x(t+1),t+1) = x^T(t+1)\, P(t+1)\, x(t+1)$

(3.3.1.9)

Replacing (3.3.1.9) in (3.3.1.8);

$$\begin{aligned} J^*(x(t),t) &= \min_{u(t)}[\, x^T(t+1)Q(t+1)x(t+1) + u^T(t)\, R(t+1)\, u(t) + x^T(t+1)\, P(t+1)x(t+1)\,] \\ &= \min_{u(t)}[(A(t)x(t)+B(t(u(t))^T Q(t+1)\, (A(t)x(t)+B(t(u(t))' + u^T(t)\, R(t+1)\, u(t) + \\ &\qquad (A(t)x(t)+B(t(u(t))^T P(t+1)\, (A(t)x(t)+B(t(u(t))\,] \\ &= \min_{u(t)}[u^T[B^TQ(t+1)B+R(t+1)]u + 2\, x^TA^TQ(t+1)Bu + x^TA^TQ(t+1)Ax \\ &\qquad + u^TB^TP(t+1)Bu + x^TA^TP(t+1)Ax + 2x^TA^TP(t+1)Bu\,] \end{aligned}$$

(3.3.1.10)

Using following matrix properties;

$$\frac{\partial}{\partial x}(x^TAx) = Ax + A^Tx \text{ and}$$

$$\frac{\partial}{\partial x}(x^TQx) = 2Qx \qquad \text{where Q is symmetric : } Q^T=Q.$$

The minimum is found by

$$J^*(x(t),t) = \min_{u(t)}[F(u(t))] \Rightarrow \left.\frac{\partial F(u(t))}{\partial u}\right|_{\min u(t)} = 0$$

since Q and R are symmetric matrices

$B^TQB = (B^TQB)^T = B^TQ^TB$, $Q^T=Q$ and $R^T=R$.

Derivation of (3.3.1.10) with respect to input u(t);

$$2[B^TQ(t+1)B + R(t+1)]\, u_{min} + 2\, x^TA^TQ(t+1)B + 2\, B^TP(t+1)B\, u_{min}$$
$$+2\, x^TA^TP(t+1)B = 0$$

$$2[\, B^T[Q(t+1)+P(t+1)]B + R(t+1)]\, u_{min} = -2\, x^TA^T[Q(t+1)+P(t+1)]B$$

$$u_{min} = u^* = -[B^T[Q(t+1)+P(t+1)]B+R(t+1)]^{-1}(x^TA^T[Q(t+1)+P(t+1)]B)$$

For a single input system, $x^TA^TP(t+1)B$ is a scalar quantity. Hence;

$x^TA^TP(t+1)B = (x^TA^TP(t+1)B)^T = B\,P^T(t+1)Ax = B\,P(t+1)Ax$, $P=P^T$

and

$S(t+1) \equiv [Q(t+1)+P(t+1)]$ then optimum input (3.3.1.11)

$u^* = -[B^T(t)S(t+1)B(t)+R(t+1)]^{-1}B^T(t)S(t+1)A\,x$.

Notice that optimum input is in the linear state feedback form

$u^* = -K\,x$ where $K=[B^T(t)S(t+1)B(t)+R(t+1)]$ is optimal state feedback gain.

$$x^T(t)P(t)x = J^*(x(t),t) = \{ [u^T[B^T(Q(t+1)+P(t+1))B + R(t+1)]u$$
$$+2x^TA^T(Q(t+1)+P(t+1))Bu + x^TA^T(Q(t+1)+P(t+1))Ax]$$
$$-x^TA^T[B^TS(t+1)B+R(t+1)][B^TS(t+1)B+R(t+1)]^{-1}B^TS(t+1)Ax \}$$

$$x^T(t)P(t)x = x^TA^TS^T(t+1)B[B^TS(t+1)B+R(t+1)]^{-1}B^TS(t+1)Ax$$
$$-2\, x^TA^TS^T(t+1)B[B^TS(t+1)B+R(t+1)]^{-1}B^TS(t+1)Ax$$
$$+x^TA^TS(t+1)Ax$$

Finally, the following equation is obtained;

$$P(t)=A^T\{ S(t+1)-S(t+1)\, B(t)[B^T(t)S(t+1)B(t)+R(t+1)]^{-1}B^T(t)S(t+1)\}A \qquad (3.3.1.12)$$

which is the matrix Riccati equation.

### 3.3.1.1 Discrete Time-Continuos Time Conversion

Any model in continuos time

$\dot{x}(t) = A(t)x(t) + B(t)u(t)$ may be transformed to the following discrete time form;

$$x(k+1) = A(k)\, x(k) + B(k)\, u(k)$$

Since derivative of x(t) may be written as

$$\dot{x}(t) = \lim_{\Delta t \to 0} \frac{x(k+1) - x(k)}{\Delta t} = A(t)x(t) + B(t)u(t)$$

$$x(t+\Delta t) = (A(t)\Delta t + I)x(t) + B(t)\Delta t u(t)$$

Discrete time models A(k) and B(k) transform into continuos time in the limit ;

$\Delta t \to 0$

$A(k) \to (A(t)\Delta t + I)$

$B(k) \to B(t)\Delta t$

and similarly for Q(k) and R(k)

$Q(k) \to Q(t)\Delta t$ and $R(k) \to R(t)\, \Delta t$

$$P(t) = (A(t)\Delta t+I)^T\{ (Q(t)\Delta t + P(t+\Delta t)) - (Q(t)\Delta t + P(t+\Delta t))\, B(t)\, \Delta t[B^T(t)\, \Delta t\Delta t\, S(t+1)B(t) + R(t+1)\, \Delta t]^{-1}B^T(t)\, \Delta t\, (Q(t)\Delta t + P(t+\Delta t))\}\, (A(t)\Delta t+I)$$

Neglecting $\Delta t^2$ and higher order terms;

$$P(t) = (A(t)\Delta t+I)^T\{ (Q(t)\Delta t + P(t+\Delta t)) - P(t+\Delta t)B(t)\, R(t+1)^{-1}B^T(t)\, \Delta t\, P(t+\Delta t)\}\, (A(t)\Delta t+I)$$

$P(t)= P(t+\Delta t)+A(t)^T P(t+\Delta t)\ \Delta t + Q(t)\Delta t - P(t+\Delta t)B(t)\ R(t+1)^{-1}B^T(t)\ \Delta t$

$P(t+\Delta t)\} + P(t+\Delta t)A(t)\Delta t$

$-[P(t+\Delta t)-P(t)]/\ \Delta t = A(t)^T P(t+\Delta t)\ + Q(t) - P(t+\Delta t)B(t)\ R(t+1)^{-1}B^T(t)\ P(t+\Delta t)\}$

$+P(t+\Delta t)A(t)$

$$-\dot{P} = Q + PA + A^T P - PBR^{-1}B^T P \qquad (3.3.1.1.1)$$

which is the Ricatti differential equation (RDE).

## 3.3.2 Continuos Time Systems

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) \qquad (3.1.2)$$

$J^*(x(t),t)= x^T(t)\ P(t)\ x(t)$ where P(t) is symmetric matrix.

$$\frac{\partial J^*(x(t),t)}{\partial t} = \frac{\partial}{\partial t}[x^T P x] = x^T \dot{P} x$$

$$\left[\frac{\partial J^*(x(t),t)}{\partial x}\right]^T = \frac{\partial}{\partial x}[x^T P x]^T = 2x^T P$$

$l(x(t), u(t), t) = u^T R u + x^T Q x$

$$x^T \dot{P} x = -\min_{u(t)}[u^T R u + x^T Q x + 2x^T P[Ax + Bu]] = -\min_{u(t)}[u^T R u + x^T Q x + 2x^T PAx + 2x^T PBu]$$

$2u^T_{min} R + 2x^T PB = 0$

$u^T_{min} = -x^T PBR^{-1}$

$u_{min} = -R^{-T}B^T P^T x$ since R,Q and P are symmetric.

$u_{min} = -R^{-1}B^T P x$

$-x^T \dot{P} x = [x^T PBR^{-1}RR^{-1}B^T P x + x^T Q x + 2x^T PAx - 2x^T PBBR^{-1}B^T P x]$

Since $x^T PAx$ is scalar and P is symmetric;

$x^T PAx = (x^T PAx)^T = x^T A^T P\ x$ then $2\ x^T PAx = x^T PAx + x^T A^T P\ x$

$-x^T \dot{P} x = x^T[PBR^{-1}RR^{-1}B^T P + Q + PA + A^T P - 2PBBR^{-1}B^T P]x = x^T[Q + PA + A^T P - PBBR^{-1}B^T P]x$

$$-\dot{P} = Q + PA + A^T P - PBR^{-1}B^T P \qquad (3.3.1.1.1)$$

which is the same RDE derived in the previous paragraph.

## 3.3.3 Time-Invariant Plants

For time-invariant A and B system with state feedback is

$$\dot{x} = Ax + Bu$$
$$u = -Kx$$
$$\dot{x} = (A - BK)x$$
$$-\dot{J} = x^T Q x + u^T R u$$
$$J \equiv x^T P x$$

The last definition for J is a Lyapunov function whose properties presented in next section. Also main reason to choose the performance index in this form is clarified

It is guaranteed that results of optimal control stabilize the system in the sense of Lyapunov stability theory.

$$\dot{J} = \dot{x}^T P x + x^T P \dot{x} = -(x^T Q x + u^T R u) \qquad (3.3.3.1)$$

$$x^T(Q + K^T R K)x = -[x^T(A - BK)^T P x + x^T P(A - BK)x] = -x^T[(A - BK)^T P + P(A - BK)]x$$

Using Cholesky factorization for R

$$R = T^T T \text{ , } (R^{-1} = T^{-1}T^{-T})$$

$$Q + K^T T^T T K = -A^T P + K^T B^T P - PA + PBK$$

$$A^T P + PA - K^T B^T P - PBK + K^T T^T T K + Q = 0$$

$$A^T P + PA - [TK - T^{-T}B^T P]^T[TK - T^{-T}B^T P] - PBR^{-1}B^T P + Q = 0 \qquad (3.3.3.2)$$

We choose,

$$TK - T^{-T}B^T P = 0$$

$$K = T^{-1}T^{-T}B^T P = R^{-1}B^T P \qquad (3.3.3.3)$$

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \qquad (3.3.3.4)$$

## 3.4. Stability of Dynamical Systems [4][80]

This section is concerned with differential equations in the form

$$\dot{x} = f(x, t) \quad \text{with } x(t_0) = x_0 \text{ where } x \in \Re^n, t \geq 0 \qquad (3.4.1)$$

The system is said to be autonomous or time-invariant, if function does not depend on t, otherwise the system is said to be nonautonomous or time-varying.

$B_h$ is defined as a ball with radius h centered at 0 in $R^n$.

The following definitions are true

locally: if all $x_0$ are in ball $B_h$.

globally: if all $x_0 \in R^n$

uniformly: if all $t_0 \geq 0$.

*Definition 3.4.1. Equilibrium Point*

$f(t,x_e) = 0$ where $x_e$ is called an equilibrium point of (3.4.1)

*Definition 3.4.2. Stability Definition of Lyapunov*

x is stable equilibrium point of (3.4.1) if there exists $\delta(t_0,\varepsilon)$ such that $|x_0| < \delta(t_0,\varepsilon) \Rightarrow |x(t)| < \varepsilon$ for all $t \geq t_0 \geq 0$ and $\varepsilon > 0$ where x(t) is the solution of (3.4.1).

*Definition 3.4.3 Uniform Stability*

If $\delta$ is independent of $t_0$ x=0 is said to be uniform stable equilibrium point of (3.4.1).

*Definition 3.4.4. Asymptotic Stability*

x=0 is said to be asymptotically stable equilibrium point of (3.4.1) if x=0 is stable equilibrium point and there exists $\delta(t_0)$ such that

$$|x_0| < \delta \Rightarrow \lim_{t\to\infty} |x(t)| = 0 .$$

*Definition 3.4.5 Uniform Asymptotic Stability(u.a.s)*

x=0 is said to be uniform asymptotically stable (u.a.s) equilibrium point (3.4.1) if x=0 is uniform stable equilibrium point of (3.4.1) and x(t) converges to 0 uniformly in $t_0$.

*Definition 3.4.6 Global Asymptotic Stability*

x=0 is said to be globally asymptotically stable equilibrium point of (3.4.1) if x=0 is asymptotically stable and $\lim_{t\to\infty} |x(t)| = 0$ for all $x_0 \in R^n$.

**3.4.1 Lyapunov Stability Theory**

*Definition 3.4.1.1 $\alpha(.),\beta(.) \in K$*

$\alpha(.),\beta(.)$: $\Re^+ \to \Re^+$ is continuos and monotonicaly increasing function with $\alpha(0)=\beta(0)=0$ .

*Definition 3.4.1.2 Locally Positive Definite Functions*

A continuos function $J(t,x): R_+ \times R^n \to R_+$ is said to be locally positive definite function (l.p.d.f) if

$J(t,0) = 0$ and $J(t,x) \geq \alpha(|x|)$ for all $x \in B_h$, $t \geq 0$.

*Definition 3.4.1.3 Positive Definite Functions*

A continuos function $J(t,x): R_+ \times R^n \to R_+$ is said to be positive definite function (p.d.f) if

$J(t,0) = 0$ and $J(t,x) \geq \alpha(|x|)$ for all $x \in R^n$, $t \geq 0$.

*Definition 3.4.4 Decresent Function*

The function $J(t,x)$ is said to be decresent if there exists a function such that

$J(t,x) \leq \beta(||x||)$ for all $x \in B_h$, $t \geq 0$.

Lyapunov second's direct method guarantees the system's stability with no need to solve the system in (3.4.1). Since, Lyapunov stability criteria investigates the only equilibrium point at x=0 , other equilibrium points should be transformed to origin. For Lyapunov's stability theory a Lypunov's function has to be defined. Any function to be tested to satisfy the properties of Lyapunov's function, is called Lyapunov's candidate function. Hence, several Lyapunov functions may be defined for a system. The Lyapunov functions satisfying definition (3.4.1.3) or definition(3.4.1.4) may be used to conclude the stability of an equilibrium point. For the stability analysis of the dynamical system $\dot{J}$, partial derivatives of J with respect to x are needed.

$$\dot{J}(x,t) = \frac{\partial x}{\partial t}\frac{\partial J}{\partial x} = f(x,t)\cdot\frac{\partial J}{\partial x}$$

Lyapunov stability theorems are defined as follow

if J(t,x) is l.p.d.f and $-\dot{J}(x,t) \geq 0$ locally then system is stable.

if J(t,x) is l.p.d.f.,decrescent and $-\dot{J}(x,t) \geq 0$ locally then system is uniformly stable.

if J(t,x) is l.p.d.f and $-\dot{J}(x,t)$ is l.p.d.f. then system is asymptotically stable.

if J(t,x) is l.p.d.f.,decrescent and $-\dot{J}(x,t)$ is l.p.d.f. then system is u.a.stable.

if J(t,x) is p.d.f and $-\dot{J}(x,t)$ is p.d.f then system is globally u.a.stable.

## 3.5. Simulation Results of Optimal Control

In this section, we will find optimal state-feedback K to minimize the cost function defined for Q and R. Simulation results with optimal state-feedback will be presented. In simulation, we will use the linear state estimator designed in section (2.6.3).

We define the following quadratic performance index to minimize;

$$J = \int_0^{\infty}(x^TQx + u^TRu)dt = \int_0^{\infty}(0.01\cdot\delta T_f^2 + 0.1\cdot\delta T_l^2 + r\cdot z_r^2)dt \qquad (3.5.1)$$

Main objective in performance index is to minimize the change in fuel temperature and temperature of coolant leaving the reactor and restrict the control rod speed. The restriction of control rod speed is achieved choosing scalar weight r very large.

For $Q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.01 & 0 & 0 \\ 0 & 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$ and r=3000, we should solve the following algebraic Riccati equation arising in section (3.3.3) for time-invariant plants. System matrix A and input matrix B in ARE (3.3.3.4) is replaced by $A_c$ and $B_c$ defined in equation (2.6.3.1) for $G_c$=0.5 to take the classical control loop into assumption.

$$A_c^TP + PA_c - PB_cR^{-1}B_c^TP + Q = 0 \qquad (3.5.2)$$

Csfopt_1.m has been written to solve ARE arising in (3.5.2) and simulate the control system illustrated in Fig.2.6.3.1. ARE is solved using function are sch.m which is a solver for stiff algebraic Riccati equations adapting Newton iteration method. Csfopt_1 has produced the following results for P and K;

```
P =1.0e+007 *

        0.0000   0.0000   0.0000   0.0000   0.0000
        0.0000   0.0001   0.0000   0.0000   0.0086
        0.0000   0.0000   0.0000   0.0000   0.0001
        0.0000   0.0000   0.0000   0.0000   0.0000
        0.0000   0.0086   0.0001   0.0000   1.9651
```

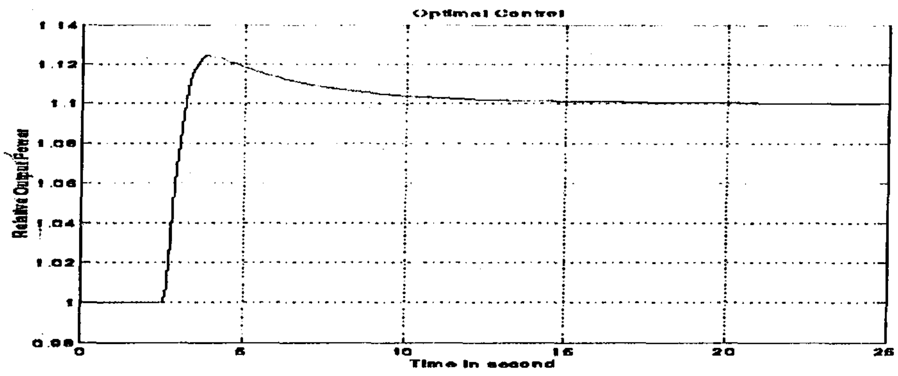
and optimal gain K is determined from equation (3.3.8)

$$K=(1/r)B_c^TP \qquad (3.5.3)$$

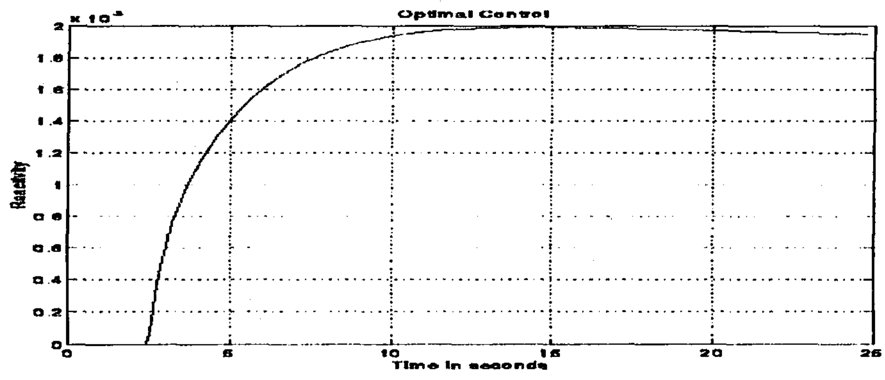
$$K = [0.0003 \quad 0.1436 \quad 0.0017 \quad 0.0002 \quad 32.7523] \qquad (3.5.4)$$

Precompensator gain v is determined using final value theorem presented in section (2.5.3) for a given optimal gain K. Nondynamical gain v compensates the input to track the demand power at steady-state.

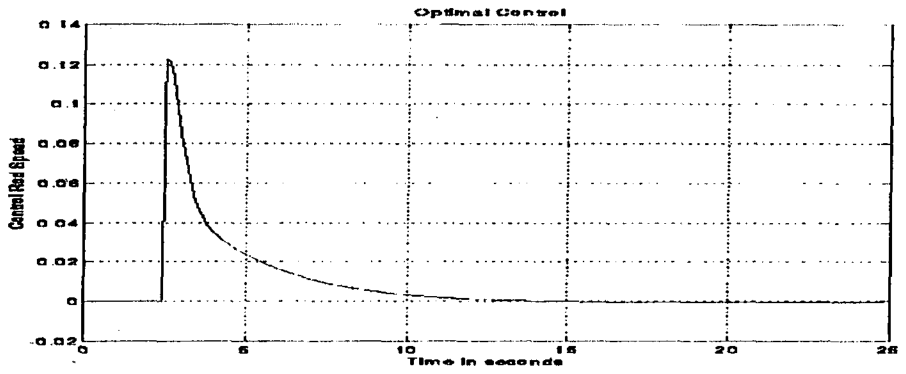
Fig 3.5.1 illustrates the output results of simulation for optimal gain K. Csfopt_1.m uses optimal gain K from (3.5.4) and nonoptimal linear observer gain H from (2.6.3.4) for $P_d$=1.1 (10% increase in demand power).



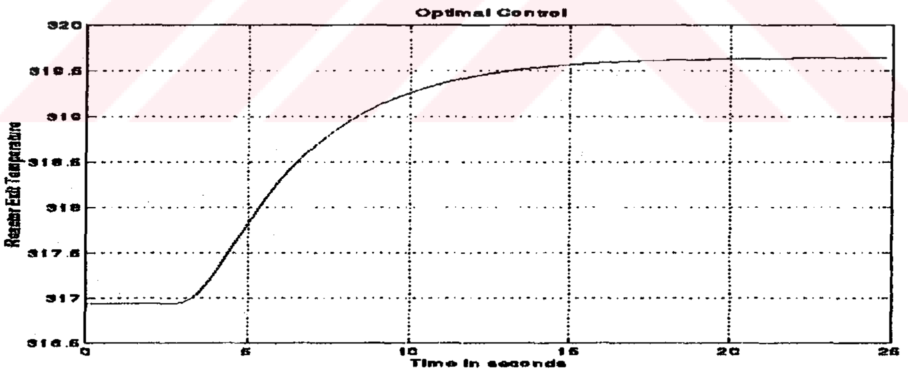
(a)



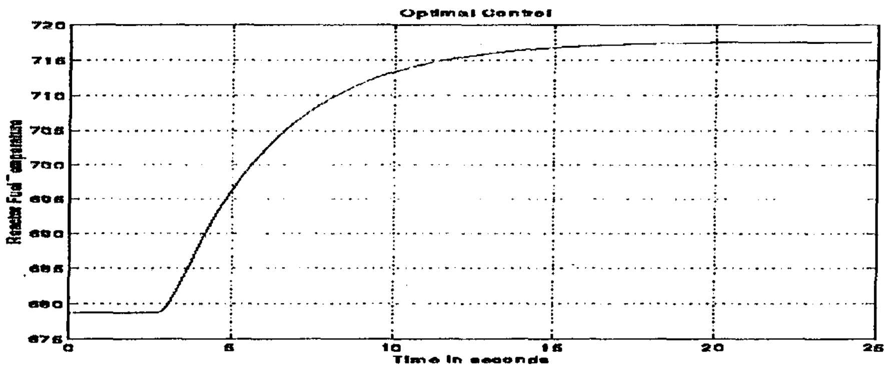
(b)


Optimal Control
Control Rod Speed
Time in seconds
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
-0.02
0
5
10
15
20
25

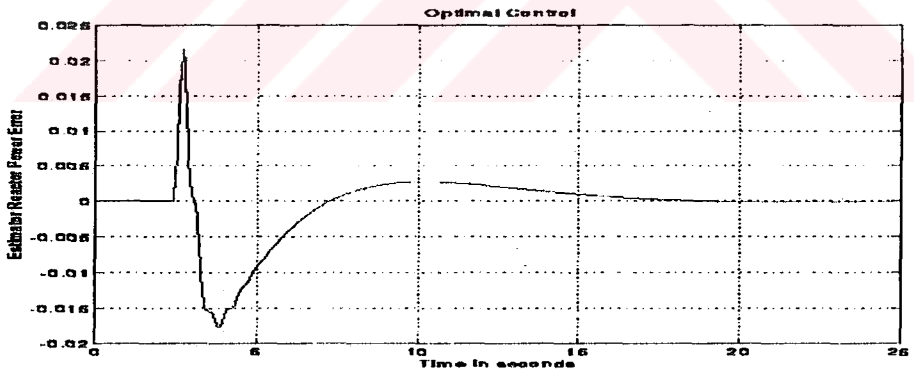
(c)


Optimal Control
Reactor Exit Temperature
Time in seconds
320
319.5
319
318.5
318
317.5
317
316.5
0
5
10
15
20
25

(d)


Optimal Control
Reactor Fuel temperature
Time in seconds
720
715
710
705
700
695
690
685
680
675
0
5
10
15
20
25

(e)


Optimal Control
Estimator Reactor Power Error
Time in seconds
0.025
0.02
0.015
0.01
0.005
0
-0.005
-0.01
-0.015
-0.02
0
5
10
15
20
25

(f)

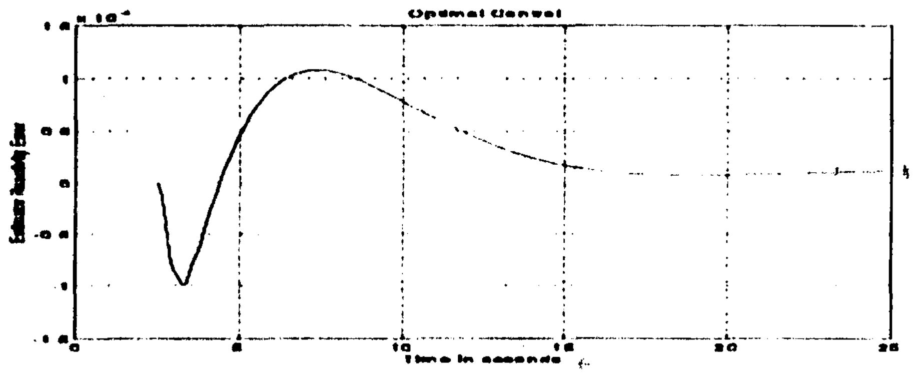

(g)

Fig 3.5.1 Optimal control results with nonlinear plant simulation (10% increase is entered to system as a demand power) a) relative output power response b) reactivity response c) control rod speed d)reactor exit temperature e) reactor fuel temperatures f) power error between the estimated power and actual value of plant g) estimator reactivity error.

## 3.6 Stiffness Arising in Optimal Control for Nuclear Power Plant

The accuracy in the solution of ARE arising in Eq.(3.5.2) is very important for the optimality and stability of control system. One of the major difficulties is encountered due to stiffness described in Section (2.5.4). Models defined by stiff equations produce the ill-conditioned matrices with relative condition numbers which are close to zero. This results in some numerical difficulties such as taking inverse. It will be shown later that Schur decomposition algorithm [84] to solve the ARE in(2.5.4) fails and produces incorrect gain K which makes the system unstable.

In Schur decomposition method proper Hamiltonian matrix for ARE in (3.5.2) is

$Z = \begin{bmatrix} A_c & -R \\ -Q & -A_c^T \end{bmatrix}$ and $U^TZU=S$ where S is the real Schur form of Z with orthogonal transformation U. U is partitioned into four parts;

$U = \begin{bmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{bmatrix}$ and to find the solution we need to take the inverse of $U_{11}$, since the solution of ARE ; $X=U_{11}^{-1}U_{21}^{T}=X^{T}= U_{21}U_{11}^{-1}$

Are_sch uses Schur method and produces the following results for ARE in (3.5.2)

$U_{11}$ =

```
 0.5565   0.0812  -0.0007  -0.6081   0.0827
-0.0011  -0.0002   0.0000  -0.0003   0.0000
-0.8182  -0.1194  -0.0027  -0.4168   0.0567
-0.0051  -0.0007  -0.0001   0.5456  -0.0858
 0.0000   0.0000   0.0000   0.0000   0.0000
```

$U_{21}$ =

```
-0.0009  -0.0066   0.9960  -0.0364  -0.0238
-0.0009   0.0064   0.0885   0.3879   0.3899
-0.0013  -0.0002  -0.0021  -0.0793   0.0828
 0.0000   0.0000   0.0115   0.0274  -0.9073
-0.1444   0.9895   0.0058  -0.0028  -0.0027
```

Relative condition number of $U_{11}$;

rcond($U_{11}$)= 6.2902e-011 which is very close to zero (badly scaled) to be taken inverse. Consequently, $U_{11}U_{11}^{-1}$ is not equal to identity matrix;

$U_{11}U_{11}^{-1}$=

```
1.0000   0.0009   0.0000   0.0000   0.0000
0.0000   1.0000   0.0000   0.0000   0.0000
0.0000  -0.0013   1.0000   0.0000   0.0000
0.0000   0.0000   0.0000   1.0000   0.0000
0.0000   0.0000   0.0000   0.0000   1.0000
```

This mistake in taking inverse leads to nonoptimal gain K. In our case, K is determined as follow;

K=1.0e+004 . [0.0006 -1.5990 0.0025 0.0015 -1.3606]

With state feedback gain K, eigenvalues of system become -63.6072, 63.607, 1.566, 1.2960 and -1.5667 which are slightly unstable.

**3.7 Conclusion**

We have designed optimal state-feedback K and simulated the SFCSE for optimal gain K determined in Eq.(3.5.4). Optimal controller denoted a similar behavior to Bang-bang control. Initially, optimal controller displayed much larger input than demand power and later turned off the offset and only produced the set value of demand power. Hence, optimal controller modifies the input signal of COC to provide time-optimal trajectories. It is seen from Fig 2.6.3.2-a and Fig 3.5.1-a that optimal K reduces the settling time to 15 seconds. It was 35 seconds for nonoptimal K determined in section (2.6.3) using pole placement method. It is also seen from Fig 2.6.3.2-d-e and Fig 3.5.1-d-e that the optimal controller improves the responses of reactor fuel temperature and reactor exit temperature. For nonoptimal K, reactor fuel temperature exceeds 720 $^{o}$C with an overshoot and settle to it's steady-state value in 45 seconds. Reactor exit temperature shows a similar dynamical behavior for nonoptimal K. Optimal controller achieves no overshoot and much shorter settling time (20 seconds).

For r=3000 with optimal gain K eigenvalues of close loop system( c.l.s.) are -63.6072, -1.5667, -1.2960, -0.3004 and -0.1294.

Different optimal gains can be determined for different scalar weight r.

For r=50, K=[0.0181 1.6038 0.0245 0.0023 269.0009], v=17.4144 and eigenvalues of c.l.s.;-63.6073, -1.4888 + 1.0222i, -1.4888 - 1.0222i, -0.1251, -1.3710.

For r=1000, K= [0.0010 0.3197 0.0037 0.0004 64.2927], v= 4.0141, eigenvalues of c.l.s.; -63.6072 -1.5294 -1.2687 -0.5256 -0.1264.

CHAPTER FOUR

# STOCHASTIC OPTIMAL CONTROL

# ROBUST CONTROLLER DESIGN

In this section we introduce the stochastic process [10][11] and stochastic optimal observer (Kalman Filter) design [12] [13] that is used in a wide variety of fields, from parameter identification to robust controller design. In chapter three we improve our controller philosophy including optimality designing optimal controllers for nuclear power plants [6]. So far, all the designs are carried out assuming that all the model parameters are correct and there exist no measurement noise. In chapter two, main design objective was to build a controller in such a way that it stabilizes the system. It was a secondary goal to improve the system's response as much as design methods allow. Although optimal control offered a method to determine the places of poles to minimize a predefined cost function, but no tool is suggested to improve the controller performance for some inaccuracies, resulting from system model or measurement.

## 4.1. Stochastic Processes

*Definition (4.1.1)*

A probability is a function P mapping some subsets $E_i$ of E into the set [1,0] with following properties [85];

$0 \leq P(E_i) \leq 1$ and $\sum_{i=1}^{n} P(E_i) =$

The joint event is defined by ABC and if the events are mutually independent.

$P(ABC...)=P(A)P(B)P(C)...$

If the events are mutually exclusive

$P(A+B+C+...)=P(A)+P(B)+P(C)$

If two events are not mutually exclusive, then

$P(A+B)=P(A)+P(B)-P(A)P(B)$

For events which are not independent, conditional probability is defined as follow;

$$(A|B) = \frac{P(AB)}{P(B)} \text{ or } (A|B) = \frac{P(B|A)P(A)}{P(B)}$$

for $A_i$, i=1,2,......,n, given that B has occurred.

$$(A_i|B) = \frac{P(B|A_i)P(A_i)}{P(B)}$$

and famous Bayes' theorem is obtained;

$$(A_i|B) = \frac{P(B|A_i)P(A_i)}{\sum P(A_i|B)P(A_i)}$$

Bayes' theorem plays the central role in probabilistic estimation theory[12][13]. The concepts stated here reflects the conventional(Boolean) logic and Bayes' theorem summarizes what we can estimate only knowing probabilities of events.

***Definition (4.1.2)***

x is a random variable(RV) and the probability distribution function is

$F_x(x)=P(X\leq x)$

and probability density function is

$$f_x(x) = \frac{\partial F_x(x)}{\partial x}$$

***Definition (4.1.3)***

E[x] denotes the expectation of x. It is also said to be mean value of x.

$$[x] = \int_{-\infty}^{+\infty} x f_x(x)dx$$

If the random variables are independent then the mean of product is also product of individual mean values of random variables;

$E[X_1X_2X_3\ X_n]=E[X_1]E[X_2]E[X_3]....E[X_n]$

***Definition (4.1.4)***

Mean square value is defined as an expectation of the square of x.

$$[x^2] = \int_{-\infty}^{+\infty} x^2 f_x(x)dx$$

***Definition (4.1.5)***

Variance of x: $\sigma^2=E[x^2]-E[x]^2$

$\sigma$ is called standard deviation

***Definition (4.1.6) Joint Distribution Functions***

The probability of the joint occurrence of two events such as A and B was called the joint probability $P(A\cap B)$ [85][86]. If the event A is the event $(X\leq x)$ and event B is the event $(Y\leq y)$ the joint Probability is called the joint distribution function of r.v. x and y

$F_{X,Y}(x,y)=P[(X\leq x)\cap P(Y\leq y)]$

***Definition (4.1.7)***

The covariance, statistical correlation between the random variables is another concept will be used frequently

$$[(x-E[x])(y-E[y])] = \int_{-\infty}^{+\infty}\int (x-E[x])(y-E[y])f_{xy}(x,y)dxdy$$

$=E[xy]-E[x]E[y]$

Correlation coefficient is the normalized covariance

$$\rho = \frac{E[xy] - E[x]E[y]}{\sigma_x \sigma_y}$$

It is obvious that if random variables x and y are independent then covariance and correlation coefficient of x and y are zero (The inverse is not true)[85].

***Definition (4.1.8) Orthogonality***

If $E[XY]=0$ then X and Y are said to be orthogonal [85].

***Definition (4.1.9) Normal or Gaussian Distribution***

The normal (Gaussian) probability density function is

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{(x-\mu_x)^2}{2\sigma^2}\right] \qquad \text{for r.v. } x$$

Mean and covariance values are only parameters to determine the Gaussian distribution [10][12][13].

For random vector x with n random variable multidimensional (multivariate) Gaussian probability density function is

$$f_n(x_1, x_2, \ldots, x_n) = \frac{1}{(2\pi)^{n/2}|P|^{1/2}} \exp\left[-\frac{1}{2}(x-m)^T P^{-1}(x-m)\right]$$

where random vector is

$x^T = [x_1 \; x_2 \; \ldots\ldots \; x_n]$

$m = E[x]$

$P = E[(x-m)(x-m)^T]$

are the mean and covariance of the vector x.

***Definition (4.1.10) Tchbycheff Inequality***

$P(|x-\mu_x| > \varepsilon) \leq \sigma_x^2 / \varepsilon^2$

where $\mu_x$ is the mean value of r.v. x and $\varepsilon$ is a positive constant[86].

***Definition (4.1.11) Chernoff Bound***

$P(x>\varepsilon) \leq \min_{t\geq 0} \exp[-t\varepsilon + \ln E\{e^{tx}\}]$

Chernoff Bound requires the probability density function unlikely Tchbycheff inequality[86].

***Definition (4.1.12) Stochastic (Random) Process***

As a random variable is a rule assigning for every outcome $\xi$ of an experiment a number $x(\xi)$, a stochastic process $x(t)$ is a rule assigning to every $\xi$ a function $x(t,\xi)$. Hence, stochastic process is a function of time and $\xi$[85]. Probability density function is

$$f(x,t) = \frac{\partial F(x,t)}{\partial x}$$

***Definition (4.1.13) Gaussian Process***

A stochastic process is called Gaussian process if its distribution functions are gaussian distributions [13][85][86]. One dimensional gaussian process is

$$f(x,t) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left[-\frac{(x-\mu_x)^2}{2\sigma^2}\right]$$

Mean and covariance values are only parameters to determine the gaussian process. Hence, the one-dimensional gaussian process is expressed by

$x \sim N(m,\sigma^2)$

and for multidimensional gaussian process

$x \sim N(m,P)$ where m, P and $\sigma$ are mean, covariance and standard deviation.

***Definition (4.1.14) Correlation Functions***

The Autocorrelation function is

$$R_{XX}(t_1,t_2)=E[x(t_1)x(t_2)]$$

and the cross-correlation function

$$R_{XY}(t_1,t_2)=E[x(t_1)y(t_2)]$$

***Definition (4.1.15) Stationarity***

A stochastic process is said to strict sense stationary (SSS) if all the probability distribution functions are invariant for time translations. It is obvious that time-invariant p.d.f describing the process is SSS.

A stochastic process called wide sense stationary (WSS) if it's mean value is constant and autocorrelation function depends only on the time difference.

$$E[X(t)] = \mu_x$$

$$E[X(t_1)X(t_2)]=R_{XX}(t_1-t_2)=R_{XX}(\tau) \quad t_1-t_2=\tau$$

Also two processes X(t) and Y(t) ere said to be jointly WSS if the following equation is satisfied;

$$E[X(t_1)Y(t_2)]=R_{XY}(t_1-t_2)=R_{XY}(\tau) \quad t_1-t_2=\tau$$

It is also obvious that SSS processes are WSS; but the inverse is not true in general.

## 4.2. Discrete-time Kalman Filter

Measurement model is defined by linear equation; [13]

$$z_k = H_k x_k + v_k \tag{4.2.1}$$

where $v_k \sim N(0,R_k)$ (zero mean, white and gaussian noise)

$\hat{x}_k$ represents the estimated state vector and state vector $x_e(k)$ is the error between actual and estimated parameters.

We are looking for a recursive filter to update $\hat{x}_k$ in the linear form;

$$\hat{x}(k+1) = K_k' \hat{x}(k) + K_k z_k \tag{4.2.2}$$

Writing equations for $x_k$ before and after the measurement, following equations are obtained

$$\hat{x}(k) = x_k + x_e(k) \quad \text{and}$$

$$\hat{x}(k+1) = x_k + x_e(k+1) \tag{4.2.3}$$

$$x_k+x_e(k+1)=K'_k(x_k+x_e(k))+K_k(H_kx_k+v_k)$$

$x_e(k+1)=K'_k x_k+K'_k x_e(k)- x_k +K_k H_k x_k+K_k v_k$

$x_e(k+1)=(K'_k x_k+K_k H_k -I)\ x_k+K'_k x_e(k)+K_k v_k$ (4.2.4)

to avoid a bias from $x_k$ let $(K'_k x_k+K_k H_k -I)=0$

$K'_k= (I-K_k H_k)$ is obtained. (4.2.5)

$x_e(k+1)= (I-K_k H_k)x_e(k)+ K_k v_k$ (4.2.6)

The cost function is defined to minimize the mean square of error to find a optimum linear gain $K_k$.

$J_k=E[x_e^T(k)x_e(k)]$ (4.2.7)

Error covariance matrix is

$P_k=E[x_e(k)\ x_e^T(k)]$ (4.2.8)

For next estimated value of the error covariance matrix becomes

$P_{k+1}=E[x_e(k+1)\ x_e^T(k+1)]$

$P_{k+1}=E\{\ [(I-K_k H_k)x_e(k)+ K_k v_k][v_k^T K_k^T+x_e^T(k)(I-H_k^T K_k^T)]\}$

$P_{k+1}=E[(I-K_k H_k)x_e(k)\ x_e^T(k)\ (I-H_k^T K_k^T)] + E[(I-K_k H_k)x_e(k)\ v_k^T K_k^T]$

$+ E[K_k v_k\ x_e^T(k)\ (I-H_k^T K_k^T)] + E[K_k v_k v_k^T K_k^T]$ (4.2.9)

and taking following equations into account;

$E[v_k\ x_e^T(k)]=0$

$E[x_e(k)\ v_k^T]=0$

$E[v_k v_k^T]=R_k$ and $P(k)=E[x_e(k)\ x_e^T(k)]$

$P(k+1) = (I-K_k H_k)P(k)\ (I-K_k H_k)^T+K_k R_k K_k^T$ (4.2.10)

$J_k=E[x_e^T(k)x_e(k)]= \text{trace}(\ P(k+1))$ (4.2.11)

The following properties are useful to find optimum $K_k$;

$$\frac{\partial}{\partial A}(\text{trace}(ABA^T)) = 2AB \qquad \frac{\partial}{\partial A}(ABA^T) = 2AB \qquad (4.2.12)$$

$$\frac{\partial J_k}{\partial K_k} = \frac{\partial}{\partial K_k}\left[(I - K_k H_k)P(k)(I - K_k H_k)^{\prime} + K_k R_k K_k^{\prime}\right] \qquad (4.2.13)$$

$$= 2\,(I-K_kH_k)P(k)\,H_k^T + 2K_kR_k = 0 \qquad (4.2.14)$$

$$K_k[R_k-H_kP(k)H_k^T]=-P_kH_k^T \qquad (4.2.15)$$

$$K_k= P_kH_k^T[H_kP(k)H_k^T+R_k]^{-1} \qquad (4.2.16)$$

$$P(k+1)= P(k) - P(k)\,H_k^T[H_kP(k)H_k^T+R_k]^{-1}H_kP(k) \qquad (4.2.17)$$

$$P(k+1)= P(k) - K_kH_kP(k) = (I-K_kH_k)\,P(k) \qquad (4.2.18)$$

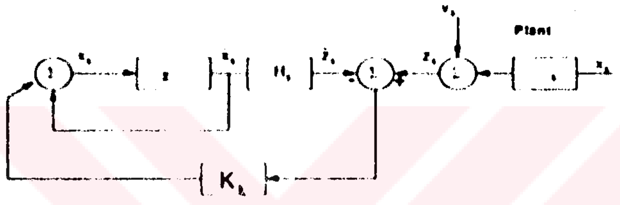


Fig 4.2.1 Discrete-Time Kalman Filter

For transformation to continous time with the measurement model

$$y(t) = C(t)\,x(t) + v(t) \qquad (4.2.19)$$

where $v(t) \sim N(0, R(t))$

$\Delta t \rightarrow 0$

$H(k) \rightarrow C(t)\Delta t$

$v(k) \rightarrow v(t)\Delta t$

$R(k) \rightarrow R(t)\,\Delta t$

$P(t+\Delta t) = P(t) - K(t)\,C(t)\Delta t P(t) = P(t) - P(t)C(t)^T\Delta t[\Delta t^2 \text{ terms} + R(t)\,\Delta t]^{-1}C(t)\,P(t)$

$P(t+\Delta t) = P(t) - P(t)C(t)^T R(t)^{-1}\Delta t C(t)\,P(t)$

$$\lim_{\Delta t \to 0} - \frac{P(t+\Delta t) - P(t)}{\Delta t} = -\dot{P} = P(t)C(t)^T R(t)^{-1} C(t)P(t) \qquad (4.2.20)$$

and Kalman gain in continuos time is

$$K(t) = \lim_{\Delta t \to 0} P(t)C(t)^T\Delta t[C(t)P(t)C(t)^T\Delta t^2 + R(t)\Delta t]^{-1} = P(t)C(t)^T R(t)^{-1}$$

(4.2.21)

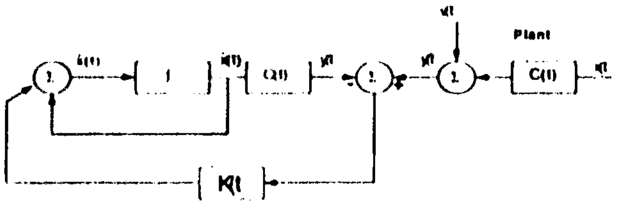


Fig 4.2.2 Continuous-Time Kalman Filter

## 4.3. Classical Output Control with Kalman Filter

In previous section we derived stochastic optimal estimator when only measurement noise exists. In general output measurements are obtained in noisy environments and gaussian process describing noise effects leads to stochastic models that match with physical realities. In this section a simple scalar Kalman filter has been used for the output of nuclear plant in Fig.4.3.1.

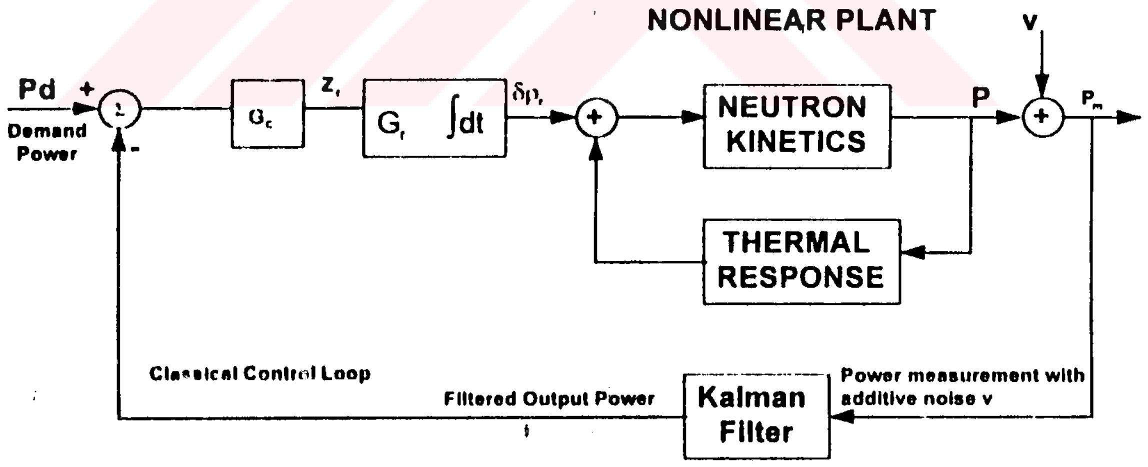
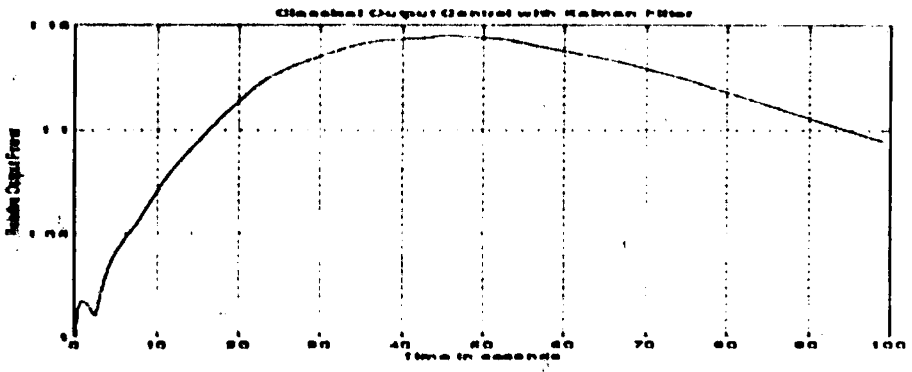
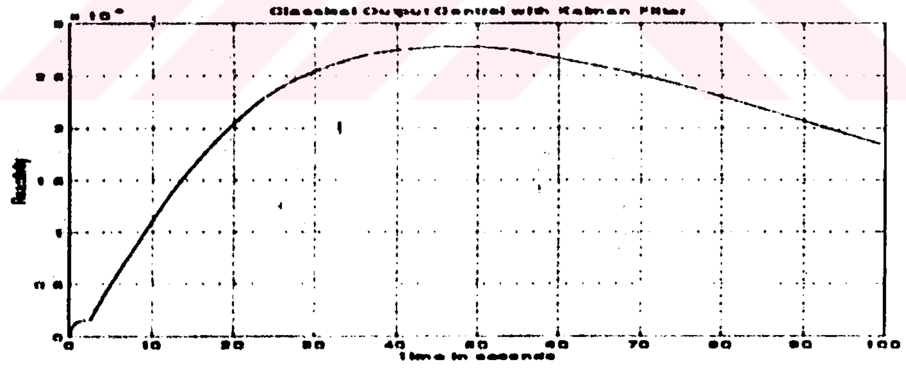


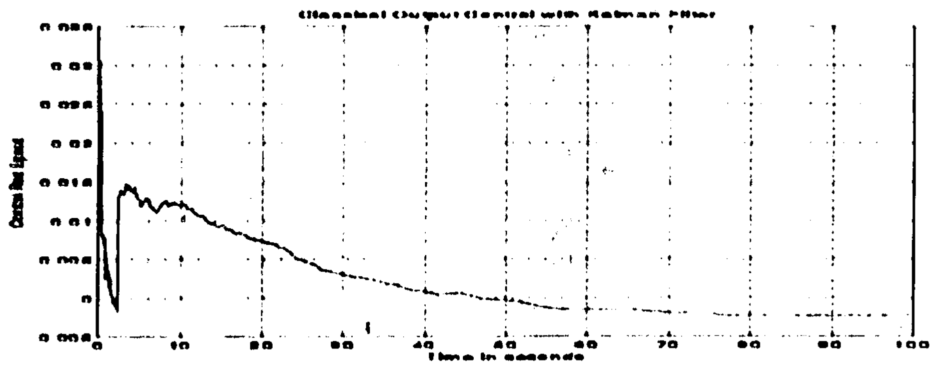
Fig 4.3.1 Block diagram of classical output control with Kalman filter


Classical Output Control with Kalman Filter
Relative Output Power
Time in seconds

(a)

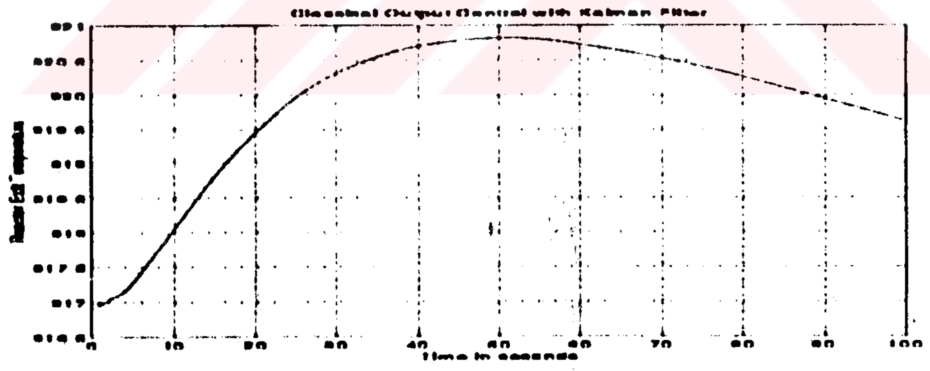

Classical Output Control with Kalman Filter
Reactivity
Time in seconds

(b)



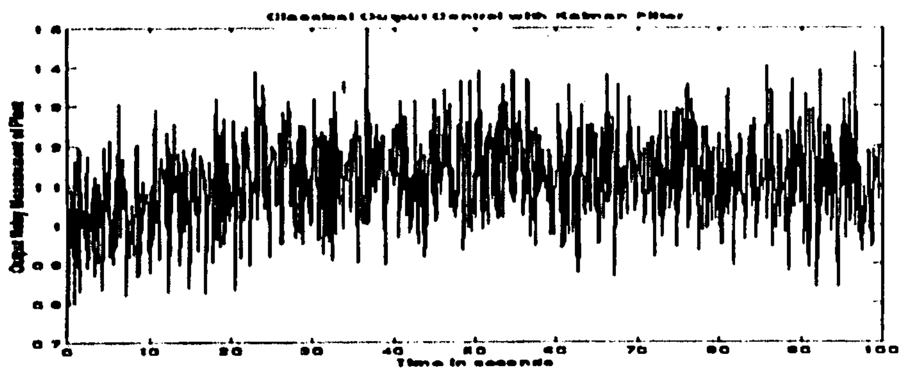
Classical Output Control with Kalman Filter
Time in seconds
0
10
20
30
40
50
60
70
80
90
100

(c)



Classical Output Control with Kalman Filter
Time in seconds
0
10
20
30
40
50
60
70
80
90
100

(d)



(e)

Fig 4.3.2 Simulation results for classical output control with Kalman filter. a) Relative output power b)Reactivity c)Control rod speed d) Reactor Exit Temperature e)Reactor Fuel Temperature e)Plant's output measurement with additive gaussian noise.

It is assumed that output neutron density measurement is obtained with additive noise v. The output of Kalman filter is used for the feedback signal to construct classical feedback loop.Nonlinear model with thermal response has been used for simulation. It is also showed in previous section that small sampling intervals for discrete time Kalman filters result in simulation of continuous time Kalman filters. Hence, plant's output with additive gaussian and zero mean noise, has been filtered using discrete time equations derived in the previous section. Fig.4.3.2. illustrates the results of COC with a simple scalar Kalman filter.

## 4.4. Stochastic Optimal Estimator Design

A system model with

$$x_{k+1} = A_k x_k + B_k u_k + w_k \qquad (4.4.1)$$

and measurement model

$z_k = H_k x_k + v_k$ (4.4.2)

where $w_k \sim N(0, Q_k)$ and $v_k \sim N(0, R_k)$ (4.4.3)

Error covariance is updated by

$$P(k+1) = A_k(I - K_k H_k) P(k) A_k^T + Q_k \quad (4.4.4)$$

with the Kalman Gain

$$K_k = P(k) H_k^T [H_k P(k) H_k^T + R_k]^{-1} \quad (4.2.16)$$

Hence error covariance updating becomes

$$P(k+1) = A_k(I - P(k) H_k^T [H_k P(k) H_k^T + R_k]^{-1} H_k) P(k) A_k^T + Q_k \quad (4.4.5)$$

Kalman Filter difference equation is obtained as

$$\hat{x}_{k+1} = A_k \hat{x}_k + K_k (z_k - H_k \hat{x}_k) + B_k u_k \quad (4.4.6)$$

Continuous time system and measurement model;

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) + w(t) \quad (4.4.7)$$

$y(t) = C(t)\, x(t) + v(t)$ where $w(t) \sim N(0, Q(t))$ and $v(t) \sim N(0, R(t)$ (4.4.8)

To obtain the continuous time results, following limit operations transforms discrete time to continuous time have been determined.

$\Delta t \to 0$

$A_k \to (A(t)\Delta t + I)$

$B_k \to B(t)\Delta t$

$H_k \to C(t)\Delta t$

$v_k \to v(t)\Delta t$

$w_k \to w(t)\Delta t$

$R_k \to R(t)\ \Delta t$

$Q_k \to Q(t)\ \Delta t$ (4.4.9)

$P_k \to P(t)$ and $P_{k+1} \to P(t+\Delta t)$ (4.4.10)

$$P(t+\Delta t)= (A(t)\Delta t+I)(I- P(t)\, C(t)^T\Delta t[C(t)\Delta t P(t)\, C(t)^T\Delta t +R\Delta t]^{-1}\, C(t)\Delta t)\, P(t)\, (A(t)^T\Delta t+I)+Q(t)\,\Delta t$$

Neglecting $\Delta t^2$ and higher order terms

$$P(t+\Delta t)= P(t)+A(t)P(t)\Delta t+P(t)A(t)^T\Delta t-P(t)C(t)^TR^{-1}C(t)P(t)\Delta t+Q(t)\,\Delta t$$

Finally, error covariance updating in continuous time is in the form of Riccati differential equation:

$$\lim_{\Delta t\to 0} -\frac{P(t+\Delta t)-P(t)}{\Delta t} = -\dot{P} = A(t)P(t)+P(t)A(t)^T - P(t)C(t)^TR(t)^{-1}C(t)P(t)+Q(t) \qquad (4.4.11)$$

with Kalman gain

$$K(t)=P(t)C(t)^TR(t)^{-1} \qquad (4.4.12)$$

and Kalman filter dynamics is

$$\dot{\hat{x}}(t) = A(t)\hat{x}(t)+K(t)(y(t)-C(t)\hat{x}(t))+B(t)u(t) \qquad (4.4.13)$$

Kalman gain K(t) obtained solving RDE minimizes the mean square error

$$J(t)=E[x_e^T(t)x_e(t)] \qquad (4.4.14)$$

It is also called minimum variance estimation.

## 4.4.1. Time Invariant Case

It is easy to show that for time-invariant models RDE (4.4.11) reduces to algebraic Riccati equation (ARE) as follow;

$$0 = AP+PA^T - PC^TR^{-1}CP+Q \qquad (4.4.1.1)$$

with time-invariant Kalman gain;

$$K=PC^TR^{-1} \qquad (4.4.1.2)$$

## 4.5. Linear Quadratic Gaussian(LQG) Stochastic Control Problem

Hence far, we have concerned with optimal control for deterministic systems and optimal estimator design for stochastic models. Optimal control of stochastic [10] systems seeks a optimum control input for uncertain models that complicate the problem. In general, stochastic control problem is related with the minimization of the cost function defined by

$$J(x(t_o),u(.),t_o) = E\left[\int_{t_o}^{T} l(x(\tau),u(\tau),\tau)d\tau \quad + m(x(T))\right] \tag{4.5.1}$$

for nonlinear plant model

$$\dot{x} = f(x,u,t) + w(t) \quad \text{with given initial condition } x(t_o) \tag{4.5.2}$$

and nonlinear measurement model

$$y(t)=c(x,t)+v(t) \tag{4.5.3}$$

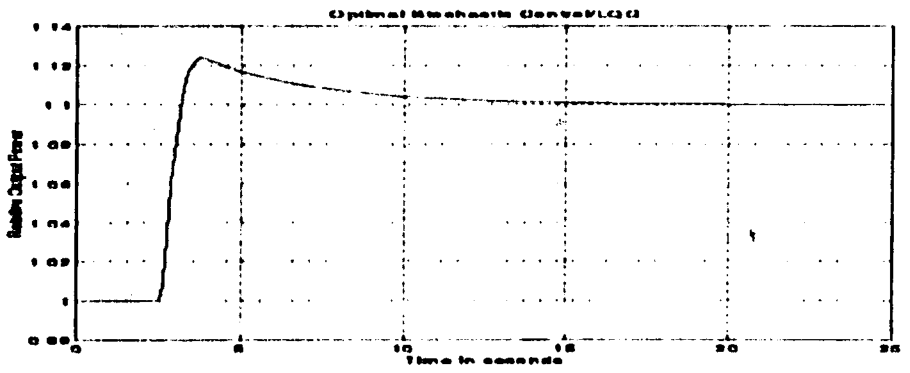
where w(t) and v(t) are stochastic processes that model the uncertainties in plant and measurement.

However LQG is a special case of stochastic optimal control if w(t) and v(t) are white, zero mean gaussian processes and performance index (cost function) has the following form:

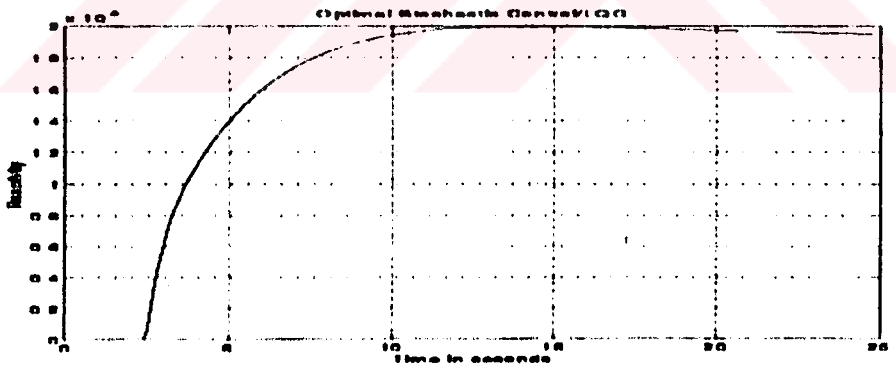
$$J(x(t_o),u(.),t_o) = E\left[\int_{t_o}^{T}(u^TRu) + x^TQx)dt + x^T(T)Gx(T)\right] \tag{4.5.4}$$

Needless to say, LQG uses the linear models for both of plant and measurement.

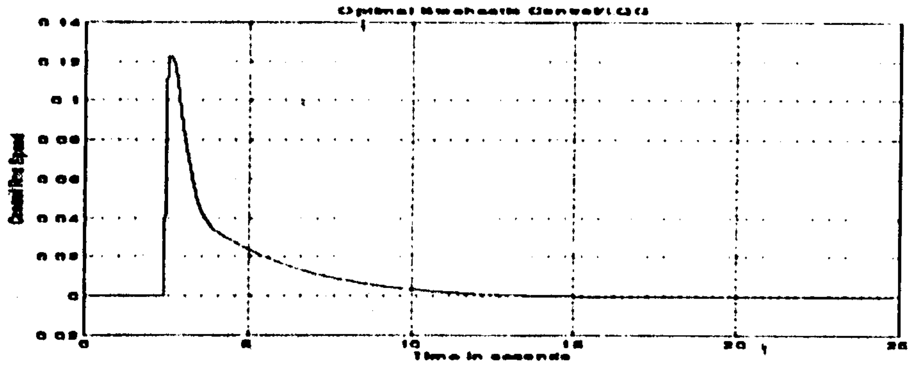
Another crucial theorem called separation principle [9] reduces this complicated problem to the combination of two design stages. One of the stages is to design a Kalman Filter using the stochastic information about the noise. At the second stage, a deterministic linear optimal controller is designed independently. This LQ regulator uses the estimated states of Kalman filter. Resultant controller minimizes the performance index defined in Eq.(4.5.4).


Optimal Stochastic Control(LQG)
Relative Output Power
1.14
1.12
1.1
1.08
1.06
1.04
1.02
1
0.98
0
5
10
15
20
25
Time in seconds

(a)

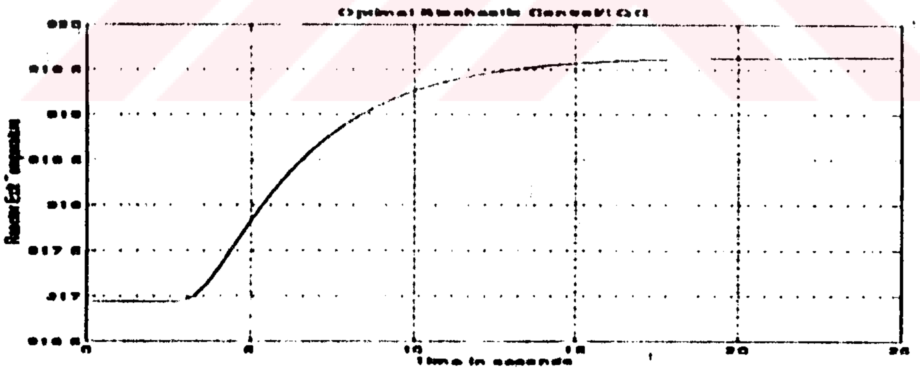

Optimal Stochastic Control(LQG)
x 10^-3
Reactivity
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
5
10
15
20
25
Time in seconds

(b)



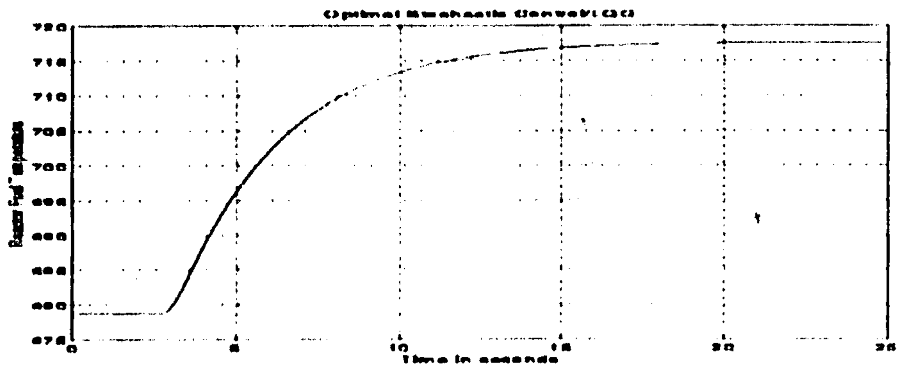
Optimal Stochastic Control(LQG)
Coolant Flow Speed
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
-0.02
0
5
10
15
20
25
Time in seconds

(c)



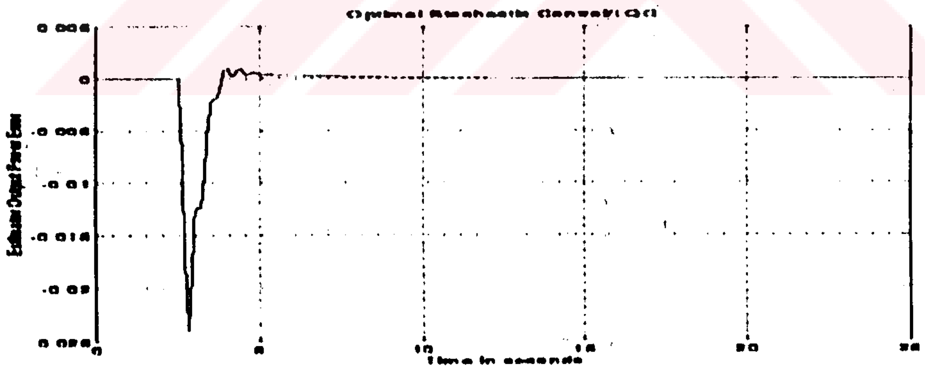
Optimal Stochastic Control(LQG)
Reactor Exit Temperature
920
919.5
919
918.5
918
917.5
917
916.5
0
5
10
15
20
25
Time in seconds

(d)



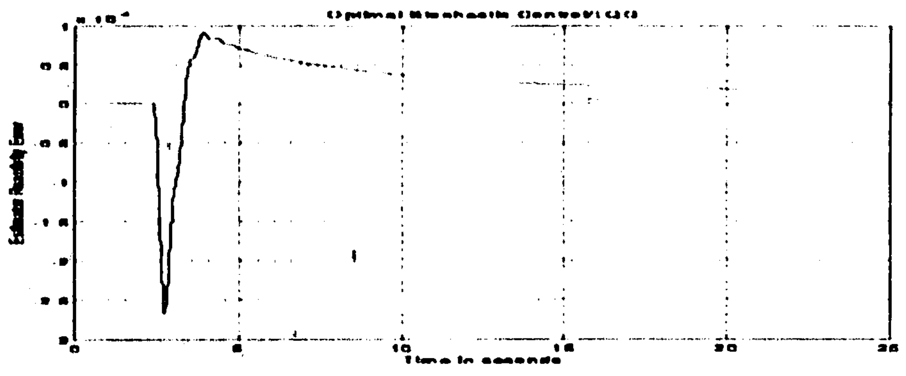
Optimal Stochastic Control(LQG)
720
715
710
705
700
695
690
685
680
675
0
5
10
15
20
25
Time in seconds

(e)



Optimal Stochastic Control(LQG)
0.005
0
-0.005
-0.01
-0.015
-0.02
-0.025
0
5
10
15
20
25
Time in seconds
Estimated Output Power Error

(f)



(g)

Fig 4.5.1 LQG : A stochastic optimal control problem simulation results for nuclear power plants. a) Reactor relative power response. b) reactivity response of system. c) Control rod speed. d)Reactor exit temperature. e) Reactor fuel temperature. (f) and (g) illustrate the error between the actual value of actual state and estimated state produced by stochastic optimal estimator (Kalman Filter) for relative output power and reactivity.

Lqg_1.m executes the function arc_2.m to solve ARE arising in Eq.(4.6.1).

```
Pe= 1.0e+008 *

        0.0093   0.0001   0.0440   0.0005   0.0001

        0.0001   0.0000   0.0159   0.0010   0.0000

        0.0440   0.0159   8.9801   0.5402   0.0006

        0.0005   0.0010   0.5402   0.0368   0.0000

        0.0001   0.0000   0.0006   0.0000   0.0000
```

and resultant H for Kalman filter is obtained as

$$H = 1.0e+003 * [-0.3095 \;\; -0.0020 \;\; -1.4650 \;\; -0.0173 \;\; -0.0024]^T$$
(4.6.2)

Fig 4.5.1 illustrates the simulation results for H and K from Eq.(4.6.2) and Eq.(3.5.4). It is seen comparing Fig.4.5.1-f-g and Fig.3.5.1-f-g that Kalman filter estimates the states in the shortest time. It takes 2-3 seconds to estimate the output power correctly for Kalman filter but it took 17 seconds for nonoptimal observer designed in section (2.6.3) from fig 3.5.1-f. Kalman filter is designed to compensate the uncertainties resulting from external disturbances or changes of model parameters. In both cases, Kalman filter estimates the states in shorter time. This improves the robustness of control system. The robustness of controllers can be examined easily changing the parameters of simulation model, while optimal state-feedback and Kalman filter are designed for the unchanged parameters. But we prefer to examine the robustness of controllers in the last chapter and end the stochastic optimal control.

CHAPTER FIVE

# AUTOMATICALLY TUNED FUZZY LOGIC CONTROLLER FOR NUCLEAR POWER PLANTS

In this chapter, we introduce fuzzy logic [15][16][17] as a new philosophy in controller design. First of all, the need to develop a new kind logic and to explain why the fuzzy logic transcends conventional logic with philosophical remarks has been included in the following sections. Control system developed [18][21][22] by using fuzzy logic is the essential spirit of the chapter. With the concept of fuzzy logic we have also introduced the intelligent control , human-like thinking and information-based systems.

## 5.1. Fuzzy Logic versus Bayesian Approach

The title of this section became the source for numerous studies. It is also subject of general discussion. Some researchers put forward that the studies with this title did not do fully justice to conventional methods. However, to assess conventional methods properly, it is worth giving this title for a comparative discussion.

Hence far, conventional logic and its natural results called Bayesian approach played the crucial role in the solutions of stochastic estimation and control problems. Therefore, these results may be classified under the title of Bayesian approach that reflects an exact view of conventional Boolean logic. It is conventional, false-true or zero-one logic that any element either is or not is a member of a group. Otherwise it is always the logical complement of the statement, there exist no third kind of classification that explains uncertain states. Multivaluedness or fuzziness is defined to overcome this problem. L.A.Zadeh known as the father of fuzzy systems, first established fuzzy fundamentals in fuzzy set theory in 1965 [24]. As an analogy, Newtonian mechanics are deterministic definitions of particle motions in physics. Under certain conditions, all the physical data may be evaluated, such as distance for a particle moving at known speed. Certainty of events is the main properties of Newtonian mechanics. However quantum mechanics appeared as another approach using Heisenberg uncertainty principle. In this case, possibility of existence of any particle in space-time is represented by probability theory. For instance an electron may exist in a cell within a probability. Whether or not it occurs, this event is modeled with probability theory. According to Kosko [25], randomness and fuzziness are not the same things although they have a lot of similarities that may be misleading to distinguish each other. An element may be member of much more than one group with possibilities. The first in quantum mechanics is randomness, the second one is fuzziness. If we search the fuzziness in physics, first of all a particle must exist in a specific space-time. An event occurs and if someone determined a physical law that the particle is electron with that percent and neutron with a different percent, proposed method would have referred to fuzziness.

## 5.2. Fuzzy Set

*Definition (5.2.1)*

X is a fuzzy set with

$X=[x_1, x_2, \ldots, x_n]=[x_i],\ i=1\ldots n$

$x_i$ is a fuzzy variable and called crisp input space $x_i \subset \Re^n$ to fuzzy set defined in A.

*Definition (5.2.2)*

Let's assume that X is a set of elements. A fuzzy subset A of X is denoted by a membership function.

$A: X \rightarrow [0,1]$

Fuzzy subset A is said to be normal if at least one element of A is 1.

The $x_i$ crisp input space $x_i \subset \Re^n$ to fuzzy set defined in A by a membership function: $\mu_A(x_i): x_i \rightarrow [0,1]$

$\mu_A(x_i)$ is membership value measures the elementhood of the ith element of fuzzy set A.

*Definition (5.2.3) Complement of A*

Complement of a fuzzy set A, denoted $\bar{\ }$, is defined by

$\bar{A}(x)=1-A(x)$

*Definition intersection*

Let's assume A and B are two fuzzy subset of X.

The intersection of these sets is equal to

$A \cap B=\min[A(x), B(x)]=A(x) \wedge B(x)$

This refers to AND logic in Boolean logic.

*Definition (5.2.4) t-norms*

t-norm is defined as a function with two arguments [87][18]

$t:[0,1]x[0,1] \rightarrow [0,1]$ satisfies the following properties

$x\ t\ 0=0$, $x\ t\ 1=x$

for $x \leq y$, $w \leq z$ $x\ t\ w \leq y\ t\ z$ nondecreasing function (monotonicity)

Also t is commutative, associative.

The intersection of two fuzzy subsets is t-norm.

Some examples for t-norms:

$x\ t\ y=\min(x,y)=x \wedge y$

$$x\ t\ y=\log_w\left[1+\frac{(w^x-1)(w^y-1)}{(w-1)}\right] \quad ,0<w<\infty \quad ,w \neq$$

$$x\ t\ y=\frac{xy}{\gamma+(1-\gamma)(x+y-xy)} \quad ,\gamma \geq 0$$

$x\ t\ y=\max[0,(\lambda+1)(x+y-1)-\lambda xy]$ , $\lambda \geq -1$

$x\ t\ y= x \bullet y$ (multiplication)

$x\ t\ y=1-\min[1,((1-x)^p+(1-y)^p)^{1/p}]$ , $p \geq 1$

It is easy to show that last t norm becomes min operation for $p \rightarrow \infty$

$\lim_{p \rightarrow \infty}\{1-\min[1,((1-x)^p+(1-y)^p)^{1/p}]\}=\min(x,y)$

*Definition (5.2.5) s-norms*

s-norm is defined as a function with two arguments [18]

$s:[0,1]x[0,1] \rightarrow [0,1]$ satisfies the following properties

$x\ s\ 0=s$, $x\ s\ 1=1$

for $x \leq y$, $w \leq z$ $x\ s\ w \leq y\ s\ z$, nondecreasing function (monotonicity)

Also s is commutative, associative.

The s-norm can be derived from the relationship

$x \; s \; y = 1 - (1-x)t(1-y)$

This relationship is equivalent to De Morgan law in Boolean logic.

Some examples for s-norms:

$x \; t \; y = \min(x,y) = x \wedge y$

$$x \, s \, y = 1 - \log_w \left[ 1 + \frac{(w^{1-x} - 1)(w^{1-y} - 1)}{(w-1)} \right] \quad ,0 < w < \infty \quad ,w \neq$$

$$x \, s \, y = \frac{xy(\gamma - 2) + x + y}{xy(\gamma - 1) + 1} \quad ,\gamma \geq 0$$

$x \; s \; y = \min[1, x+y+\lambda xy] \quad , \lambda \geq -1$

$x \; s \; y = x+y-xy$

$x \; s \; y = \min[1, (x^p+y^p)^{1/p}] \quad , p \geq 1$

It is easy to show that last s-norm becomes min operation for $p \to \infty$

$\lim_{p \to \infty} \{ \min[1, (x^p+y^p)^{1/p}] \} = \max(x,y)$

*Definition (5.2.6) The size (cardinality) of A*

The cardinality of a fuzzy set A, M(A) is defined by

$$(A) = \sum_{i=1}^{n} \mu_A(x_i)$$

*Definition Fuzzy Hamming distance*

The distance between the fuzzy sets A and B in $\ell^p$:

$$\ell^p(A,B) = \sqrt[p]{\sum_{i=1}^{n} |\mu_A(x_i) - \mu_B(x_i)|^p}$$

*Definition (5.2.7) Possibility:*

The possibility of V is B given V is A:

$$\text{poss}[B/A] = \max_x [A(x) \wedge B(x)]$$

It is obvious from definition that

$$\text{poss}[B/A] = \text{poss}[A/B].$$

*Definition (5.2.8) Certainty*

Certanity is denoted and defined by

$$\text{cert}[B/A] = 1 - \text{poss}[\bar{B}/A]$$

and following inequality is satisfied

$$cert(B/A) \le poss[B/A]$$

## 5.3. Fuzzy Logic Systems

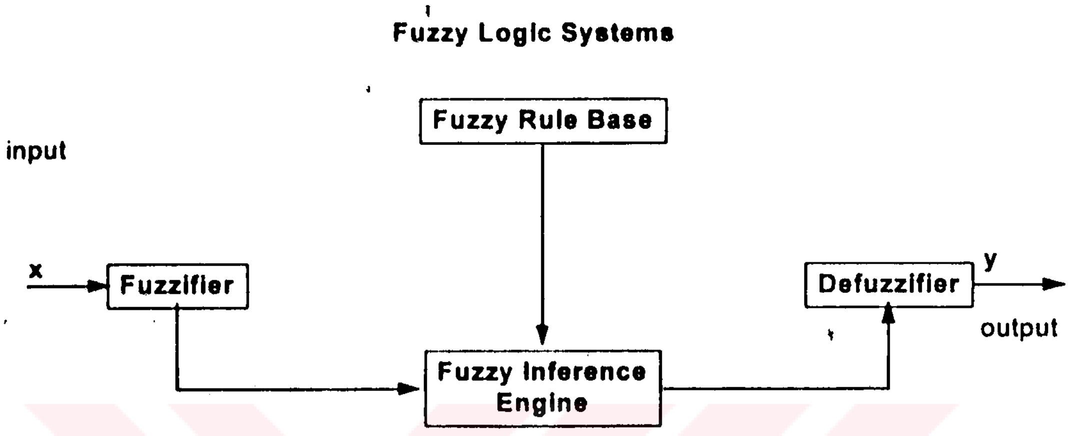


Fig 5.3.1 Basic Configuration of fuzzy logic systems.

The block diagram shows input-output relationship of the machine with a transfer function.

The fuzzifier performs a mapping from observed crisp input X to fuzzy set defined a membership function $\mu_A$. $X \rightarrow [0,1]$.

Fuzzy rule consists of a set of linguistic IF-THEN rules. Jth fuzzy rule $R_j$ is defined by

$R_j$:IF $x_1$ is $A_1^j$ and $x_2$ is $A_2^j$ and ... and $x_n$ is $A_n^j$, THEN y is $B^j$.

where $j=1,2,\ldots,N$, $x_i(i=1..n)$

Output of j th fuzzy rule is a fuzzy set defined in $B^j$. IF statement is the ANTECEDENT and THEN portion is the CONSEQUENT.

The fuzzy inference machine is a decision maker mapping some fuzzy sets from fuzzy rules into another fuzzy set. The output of fuzzy inference machine is transformed to the real world as an output of fuzzy logic system by the defuzzifier.

The Defuzzifier uses the several methods to provide the nonfuzzy output set (crisp) $y^*$. Mean of maxima (MOM) and center of gravity are commonly used procedures.

## 5.4 Fuzzy Logic Control

In fuzzy logic control [18][21][22][23], knowledge is represented by the following if-then rules $R_1$: IF $E^1$ and $DE^1$ Then $U_1$

$R_2$: IF $E^2$ and $DE^2$ Then $U_2$

$R_j$: IF $E^j$ and $DE^j$ Then $U_j$

where j=1..N for N fuzzy rules.

They represents the membership values of the error and change in error of the output of the controlled process. For n fuzzy set partitioning $A_1, A_2, \ldots, A_n$ we can derive nxn fuzzy rules. So total number of fuzzy rules must satisfy

$N \leq n^2$

$U_j$ is the control input of the j-th rule defined in fuzzy set. For multi-input multi-output (MIMO) systems error E, change in error DE and control input U are defined in set

$E=\{e_1, e_2, \ldots e_{n1}\}$, $DE=\{de_1, de_2, \ldots de_{n2}\}$ $U=\{u_1, u_2, \ldots u_{n1}\}$

and for every control input u, fuzzy inference machine produces proper outputs in fuzzy set defined by the fuzzy rules.

For the AND operation in IF part, any t-norm such as min operation or multiplication, can be used.

$(E^j \text{ and } DE^j) = \max(E^j, DE^j)$ or

$(E^j \text{ and } DE^j) = E^j \bullet DE^j$

### 5.4.1 Control with Linguistic Variables

Fuzzy sets A can be defined in linguistic variables [18]. Every fuzzy set A is represented by a linguistic quantity. According to Kosko[25] these linguistic variables are

| | |
|---|---|
| NL: | Negative Large |
| NM: | Negative Medium |
| NS: | Negative Small |

| | |
|---|---|
| ZE: | Zero |
| PS: | Positive Small |
| PM: | Positive Medium |
| PL: | Positive Large |

## 5.5 Classification of Fuzzy Logic Controllers

We mainly classify the fuzzy logic controller into two groups. These are (A)non-adaptive and (B)adaptive fuzzy logic controllers.

(A) Non-adaptive fuzzy logic systems consist of predefined time-invariant rules and membership functions. These are knowledge bases defined in fuzzy logic controller. In one type, IF part and Then part of fuzzy rules are in the linguistic forms (fuzzy sets) as follows

Type A1: Rule : IF $e_k$ is NL and $de_k$ is PL THEN du is PL

Then part can also be in polynomial form of e and de. It is called Sugeno type output [19][20].

Type A2: Rule : IF $e_k$ is NL and $de_k$ is PL THEN $u=k_1e_k+k_2de_k+k_3$

(B) Adaptive fuzzy logic systems

Membership functions and fuzzy rules are automatically tuned according to supervisory data ( input output data of plant).

Type B1: Membership functions are knowledge-based but fuzzy rules are defined adaptively. These are

B1.1:The adaptive fuzzy associative memories(AFAM) in Kosko's work[25].

B1.2:The fuzzy logic controllers as cerebellum model articulation controller (CMAC)[38][18]. In CMAC, the fuzified data is entered to a neural networks that finds fuzzy rules and defuzified output of the controller properly.

B1.3: Consequents are Sugeno type polynomial function of crisp inputs. The parameter of consequents are tuned using either Kalman filters or least square estimation according to a supervisory signal to provide the desired response for fuzzy logic controller [14].

Type B2 Membership functions and fuzzy rules are determined in adaptive fuzzy logic controller structures. Fuzzy Neural Networks (FNN) are this kind of structures whose membership functions and fuzzy rules are automatically tunable.

## 5.6. Fuzzy Logic Controller (FLC) for Nuclear Power Plant

In this section, we introduce design of a fuzzy logic controller defined as B1.3 type in the previous section. Fig 5.3.2 illustrates the scheme of fuzzy logic controller application to nuclear power plants. As a general statement, FLC illustrated in Fig 5.3.2 can be used for the every kind of nonlinear plant without any restriction.

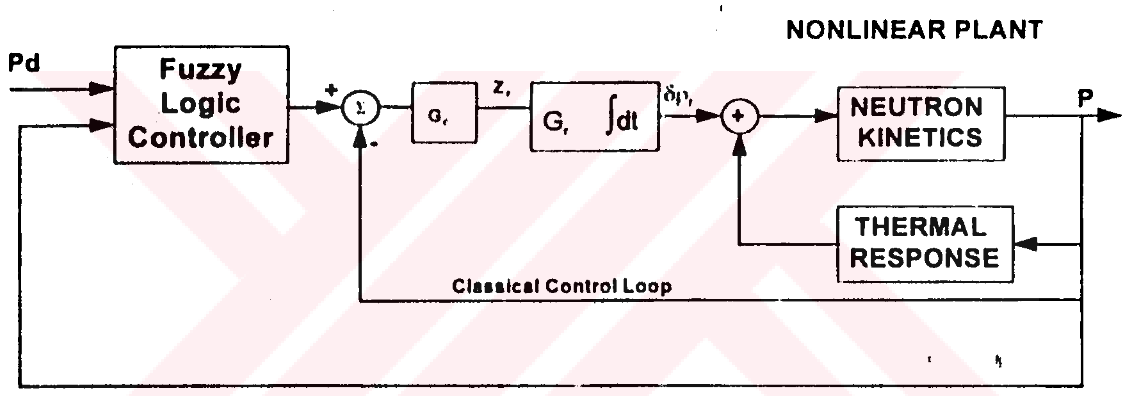


Fig 5.3.2 Fuzzy Logic Control of Nuclear Power Plant

In our example, output power of nonlinear plant is unique output to be controlled. It is also possible to control other state variables such as fuel or exit temperatures. In this case, an observer is used to estimate the temperatures. Conventional control loop is saved but whether or not it is saved fuzzy logic controller can be designed for both cases. FLC produces the inputs to plant with conventional loop. Fuzzy rules are tuned using a supervisory signal provided by a LQG control developed in the chapter four. The tunning process can be carried out in several ways. One of the most practical methods is the least square estimation and another is the mean square estimation (Kalman Filter) to tune the fuzzy rules. In the later sections, we will focus on the tunning methods more deeply. As it is seen from scheme, main structural difference between the optimal controllers and FLC is that FLC uses only plant's output power while the optimal controllers need estimated values of state variables.

Five fuzzy sets $A_i$ I=1,2,..,5 ;$A_1$=large negative (LN), $A_2$=small negtive(SN), $A_3$= zero (ZE), $A_4$= small positive (SP), $A_5$= large positive(LP) have been used in FLC design. Outputs of fuzzy rules are in the polynomial form defined by Sugeno as follow;

Rule1: if E(k) is $A_1$ and DE(k) is $A_1$ then $u_1(k)= c_1^0+c_1^1 E(k)+c_1^2 DE(k)$

Rule2: if E(k) is $A_1$ and DE(k) is $A_2$ then $u_2(k)= c_2^0+c_2^1 E(k)+c_2^2 DE(k)$

.....

Rule25: if E(k) is $A_5$ and DE(k) is $A_5$ then $u_{25}(k)= c_{25}^0+c_{25}^1 E(k)+c_{25}^2 DE(k)$

Where error for the kth sample $E(k)=P(k)-P_d(k)$; P and $P_d$ are process output(plant's power) and demand power, respectively.

The change in error is defined by DE(k)= E(k)-E(k-1).

Maximum twenty five fuzzy rules can be defined for five fuzzy sets. Knowledge based membership functions have been defined by J.Dombi[88] in two functions. The first one is monotonically increasing function defined in [a,b]:

$$\mu(x)=\frac{(1-v)^{\lambda-1}(x-a)^{\lambda}}{(1-v)^{\lambda-1}(x-a)^{\lambda}+v^{\lambda-1}(b-x)^{\lambda}}$$

and the second is the monotonically decreasing function:

$$\mu(x)=\frac{(1-v)^{\lambda-1}(b-x)^{\lambda}}{(1-v)^{\lambda-1}(b-x)^{\lambda}+v^{\lambda-1}(x-a)^{\lambda}}, \qquad x\in[a,b] \qquad (5.6.1)$$

where λ and v are the sharpness and inflection points of S-shaped functions. The inputs error E and change in error DE to FLC are fuzzified by μ(E) and μ(DE) defined in 5.6.1. Every fuzzy rule is weighted according to AND function of membership values for E and DE. This is processed using either min or multiplication (or any t-norm). We prefer to use the multiplication to determine the weights of fuzzy rules. Output of the FLC is the weighted avarage of outputs for every fuzzy rule as defuzzification process.

$$u^{*}=\frac{\sum_{i=1}^{5}\sum_{j=1}^{5}\mu_{ij}(E)\mu_{ij}(DE)u_{((i-1)\times 5+j)}}{\sum_{i=1}^{5}\sum_{j=1}^{5}\mu_{ij}(E)\mu_{ij}(DE)}$$

### 5.6.1. Kalman Filter Approach to Identify The Paramaters of Fuzzy Rules

We start the work defining the state-space model of adaptive fuzzy logic controller. The state vector contains the unknown parameters $c_i^k$ in the outputs of fuzzy rules. Unknown parameters appears in the consequent $c_i^k$ $i=1,2,\ldots,n.,k=1,2$ . The ith fuzzy rule output is

$u_i= c_i^0+c_i^1 E+c_i^2 DE$

and state-vector is defined by

$$x=\begin{bmatrix}c_1^1 & c_2^1 & c_3^1 & . & c_n^1 & c_1^2 & c_2^2 & c_3^2 & . & c_n^2\end{bmatrix}^T$$

System model

$x_{k+1} = x_k + w_k$

measurement model

$u^{*}=y=[B_1E(k)\ B_2E(k)\ldots\ldots B_nE(k)\ B_1DE(k)\ B_2DE(k)\ldots\ldots B_nDE(k)]\ x_k+v_k+c^0$

for n=25, $B_1, B_2 \ldots\ldots B_{25}$ represent the weighted avarages of fuzzy rules.

$$B_{(i-1)\times 5+j}=\frac{\mu_{ij}(E)\cdot\mu_{ij}(DE)}{\sum_{i=1}^{5}\sum_{j=1}^{5}\mu_{ij}(E)\mu_{ij}(DE)}$$

$w_k$ and $v_k$ are white, zero mean gaussian process and measurement noises.

From the results derived in chapter four with $A_k=I_{2n\times 2n}$ and measurement model:

$z_k = H_k\ x_k + v_k$ (4.4.2)

where $w_k \sim N(0,Q_k)$ and $v_k \sim N(0,R_k)$ (4.4.3)

To minimize the mean square error

$J_k=E[x_e^T(k)x_e(k)]$

Error covariance is updated by

$$P(k+1)= (I-K_kH_k)\, P(k)+Q_k$$

with the Kalman Gain

$$K_k= P(k)H_k^T[H_kP(k)H_k^T+R_k]^{-1} \qquad (4.2.16)$$

Hence error covariance updating becomes

$$P(k+1)= (I- P(k)H_k^T[H_kP(k)H_k^T+R_k]^{-1}H_k)\, P(k)+Q_k$$

With the following Kalman Filter difference equation, new estimated parameters of fuzzy rules are easily obtained

$$\hat{x}_{k+1} = \hat{x}_k + K_k(z_k - H_k\hat{x}_k)$$

$z_k$ is the desired output of FLC. This signal is produced by a supevisory conroller (In this work, we shall use LQG designed in chapter four as a supervisory controller).

### 5.6.2. Least Square Estimation of Fuzzy Rules

Least square estimation [89] is another tool which may be used instead of mean square estimation (Kalman Filter)[13]. In the least square estimation, following cost function is minimized;

$$J_k = (z_k - H_k\hat{x}_k)^T(z_k - H_k\hat{x}_k) = \|(z_k - H_k\hat{x}_k)\|_2 \qquad (5.6.2.1)$$

It is easy show that to satisfy

$\frac{\partial J_k}{\partial \hat{x}} = 0$ with positive semidefinite Hessian of J $\left|\frac{\partial^2 J_k}{\partial \hat{x}^2}\right| \geq 0$

$$\hat{x}_k = (H_k^TH_k)^{-1}H_k^Tz_k = H_k^{\#}z_k$$

where $H^{\#}$ is pseudo-inverse (or generalized inverse or Moore-Penrose inverse)

If $H^TH$ is non-singular then pseudo-inverse can be determined easily. Otherwise, singular value decomposition is used to find the solution in least square estimation.

*Definition Orthogonality*

A matrix Q $\varepsilon\Re^{m\times m}$ is said to be orthogonal if $Q^TQ=I$.

*Theorem Singular Value Decomposition(SVD)*

If a real matrix $A \in \Re^{m \times n}$ then there exist orthogonal matrices

$U=[u_1,u_2,\ldots,u_m] \in \Re^{m \times m}$ and $V=[v_1,v_2,\ldots,v_m] \in \Re^{n \times n}$

with the following equation

$U^T A V = \mathrm{diag}(\sigma_1,\ldots\ldots,\sigma_p) \in \Re^{m \times n}$ where $p=\min(m,n)$ and $\sigma_1 \geq \sigma_2 \geq \ldots\ldots \geq \sigma_p \geq 0$

The $\sigma_i$ are the singular values of A.

*Theorem Least Square Minimization by SVD*

Let's assume $U^T H V = \Sigma$ is the SVD of matrix $H \in \Re^{m \times n}$ with $r=\mathrm{rank}(H)$

To minimize the (5.6.2.1)

$$\hat{x} = \sum_{i=1}^{r} \frac{u_i^T z}{\sigma_i} v_i \quad .$$

## 5.6.3 Tuning Mechanism

A supervisory signal is used to tune the fuzzy rules in FLC. This supervisory signal is produced by a reference controller. In our case, we have used optimal controller LQG designed in chapter four for the reference controller. The simulation results of plant for $P_d=0.6$(%40 decrease in demand power) are accommodated in the fuzzy rule identification. For this purpose, Fuzzyr_1.m has been coded to tune the fuzzy rules according to the output of a reference controller. Signals accompanied with Fuzzyr_1.m are the demand power$Y_d$ ,control input of reference controller$Y_c$, output response of plant $Y_p$ . Yc is the modified control input of LGQ to modify the dynamical response of plant. Fuzzy rules determined by Fuzzyr_1 are used in Fc_1.m to simulate the FLC for nonlinear plant. Fig.5.6.3.1 shows the tuning mechanism with LQG.

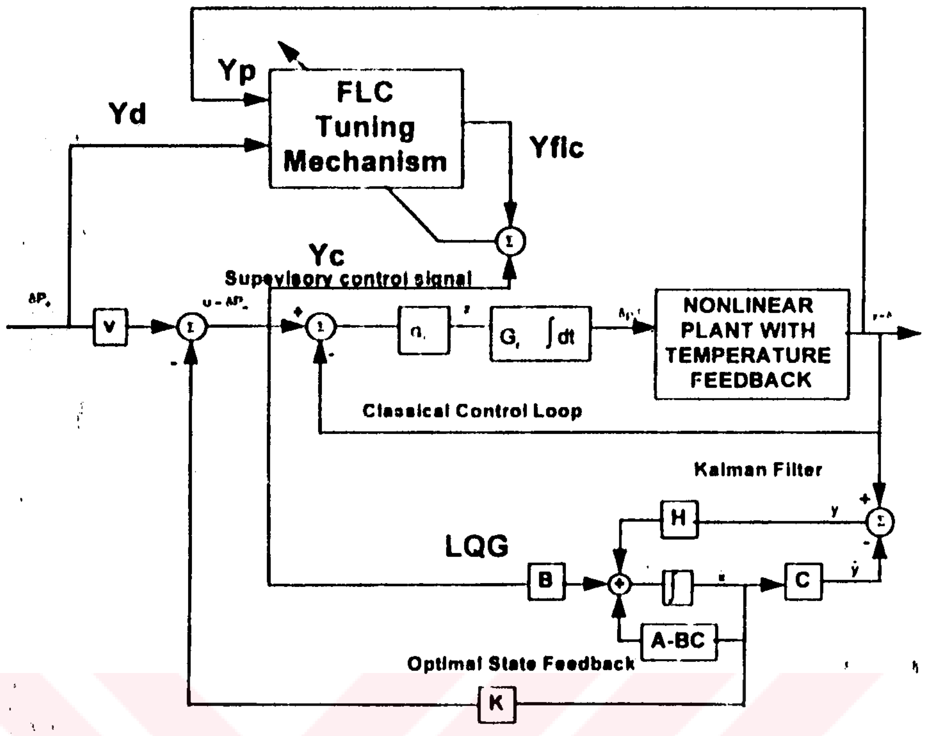
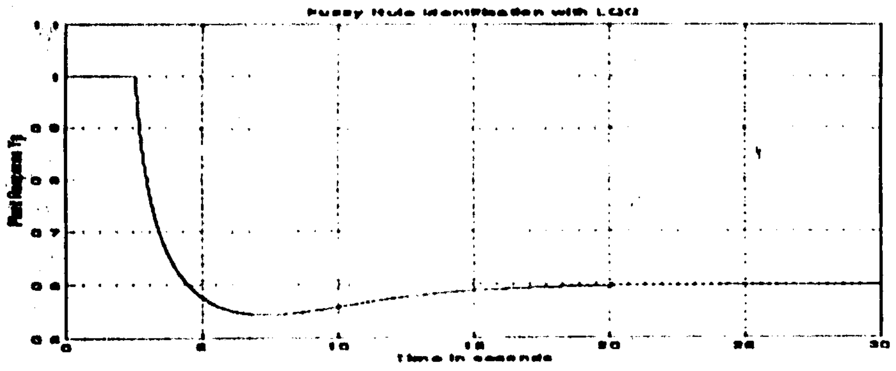


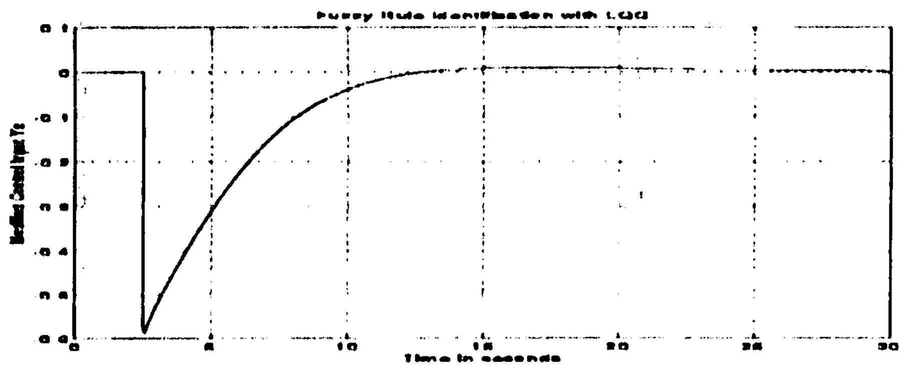
Fig 5.6.3.1 FLC Tuning Mechanism

### 5.6.4 Identification of Fuzzy Rules

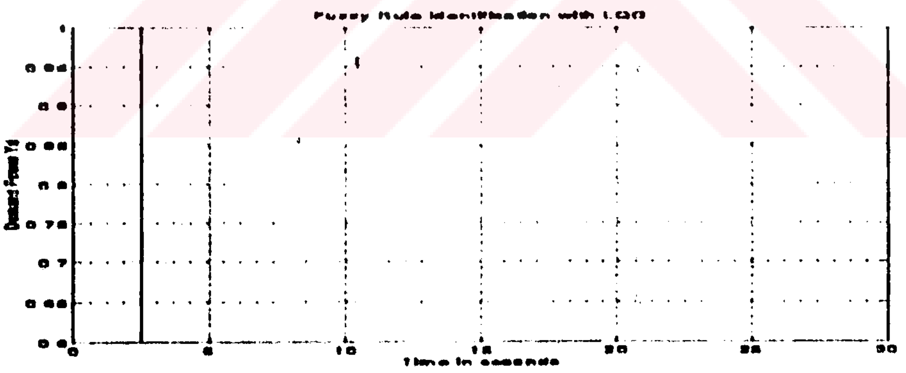
We need plant response $Y_p$, control input of a reference controller $Y_c$, and demand power $Y_p$.



(a)



(b)



(c)

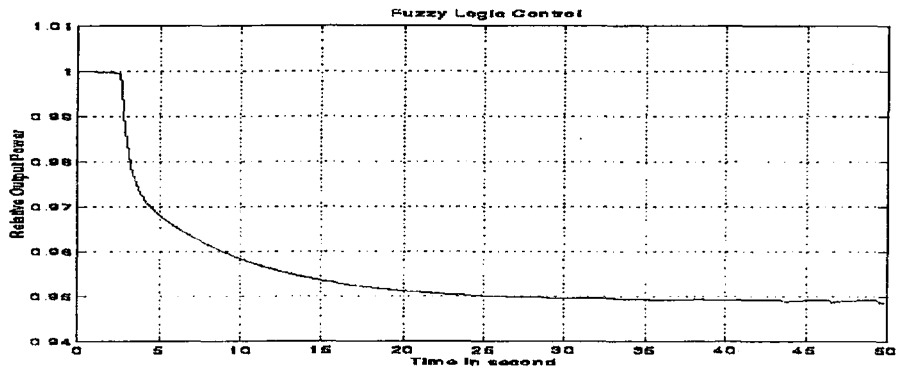
Fig 5.6.4.1 Results of LQG are a)plant response(relative neutron density) $Y_p$ b)modified control input $Y_c$ c)demand power $Y_d$.

Signals illustrated in fig 5.6.4.1 are used in Fuzzyr_1.m and the following parameters are determined for fuzzy rules in FLC;

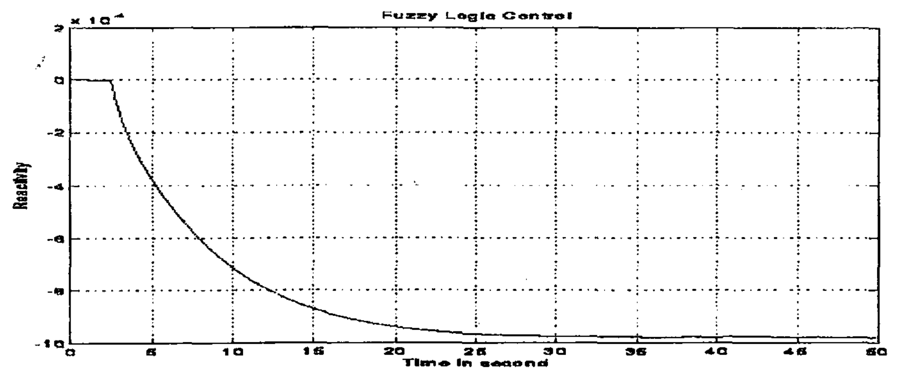
| Rules | $c_i^1$ | $c_i^2$ |
|---|---|---|
| (1.0e+004) * | | |
| 1 | 0 | 0.0000 |
| 2 | -0.0013 | -0.1048 |
| 3 | 0.0002 | 0.1313 |
| 4 | 0.0014 | -0.7032 |
| 5 | 0.0000 | 0.0000 |
| 6 | 0.0000 | 0.0000 |
| 7 | 0.0367 | 2.6936 |
| 8 | -0.0017 | -0.9410 |
| 9 | -0.1419 | 0.0528 |
| 10 | 0.0000 | 0.0000 |
| 11 | 0.0000 | 0.0000 |
| 12 | 0.0089 | 0.2052 |
| 13 | -0.0005 | 0.0105 |
| 14 | 0.1792 | -0.0005 |
| 15 | 0.0000 | 0.0000 |
| 16 | 0.0000 | 0 |
| 17 | -0.1635 | 0.0377 |
| 18 | 0.0696 | 0.0910 |
| 19 | 0.0000 | 0.0000 |
| 20 | 0.0000 | 0 |
| 21 | 0.0003 | 0.0200 |
| 22 | -0.0001 | 0.0126 |
| 23 | 0.0000 | 0.0695 |
| 24 | 0.0000 | 0 |
| 25 | 0.0000 | 0 |

## 5.7. Simulation of FLC for Nuclear Power Plant

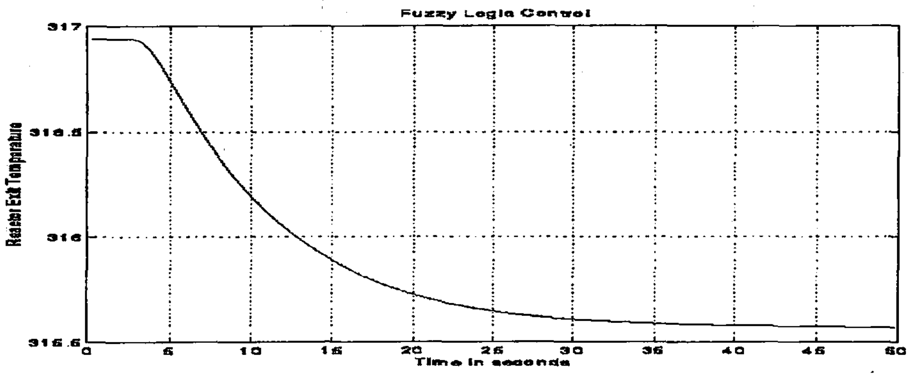
Fc_1.m has been run to simulate the control system illustrated in fig 5.3.2 taking the parameters determined in the previous section. Fig 5.7.1 illustrates the simulation results for FLC.



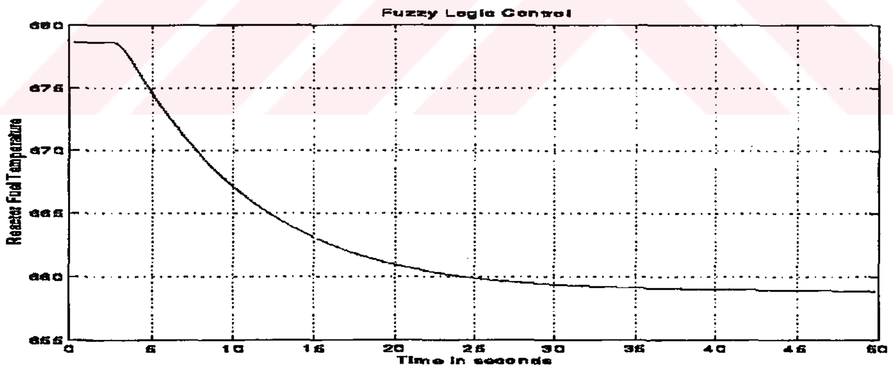
(a)



(b)



(c)



(d)

Fig 5.7.1 Fuzzy logic control simulation results for a) relative output power b) reactivity response c) reactor exit temperature and d) reactor fuel temperature.

## 5.8 Conclusion

In this chapter, we have designed a fuzzy logic controller(FLC) for nuclear power plant. Hence far, control methodologies studied previously were linear model based approaches. Among them SFCSE with pole placement and optimal control have been studied extensively in the previous chapters. FLCs have appeared as a control methodology which can be applied to any nonlinear plant without knowing its mathematical model. Essentially, main problem arising in FLCs is to identify the fuzzy rules. To achieve this goal, we have presented several methods to tune FLC properly. Some of these methods to identify the parameters of fuzzy rules, are mean square estimation (Kalman Filter), least square estimation (LSE) , least mean square LMS algorithm and cell state space algorithm. Cell state space algorithm was developed for tuning fuzzy rules [54]. Also LMS (developed by B.Widrow[26][34]) is another algorithm using gradient descent for fuzzy rule identification. We have preferred the LSE algorithm and identified the parameters $c^1$ and $c^2$ which appear in the consequent part of fuzzy rules. It is seen from the simulation results that FLC have learned a control surface produced by LQG as a reference controller. The control surface learned by FLC is illustrated in fig 5.8.1. For a structural comparison, FLC in fig 5.3.2 uses only the output of plant meanwhile optimal controller requires the states of plant.

FLC may be assumed as a combination of N PD (proportional derivative) controllers. Objective of intelligent control is to develop the methods combining previously designed conventional controllers or experiences about the plant in such a way that overall performance of control system is improved. FLC supervises the different PD controllers using fuzzy rules which are determined from the information of plant. Each PD controller is weighted by a fuzzy rule according to the dynamical behavior of plant. Therefore, FLCs are powerful information-based intelligent control tool.

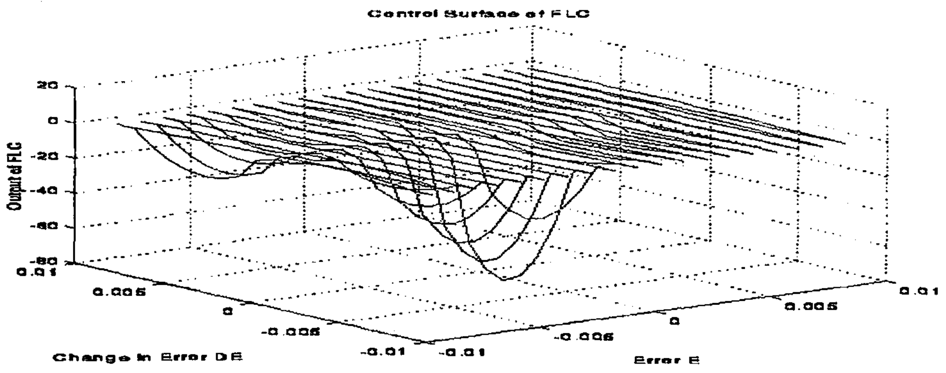


Fig 5.8.1 Control surface of FLC for parameters determined in section (5.6.4). To achieve this control surface, fuzzy rules are tuned for the supervisory signal which is produced by LQG designed in the chapter four.

CHAPTER SIX

# ARTIFICIAL NEURAL NETWORKS AND NEUROCONTROL OF NUCLEAR POWER PLANTS

In this chapter we introduce the neural networks and their applications to nuclear power plants. Necessary fundamentals will be given with an overview.

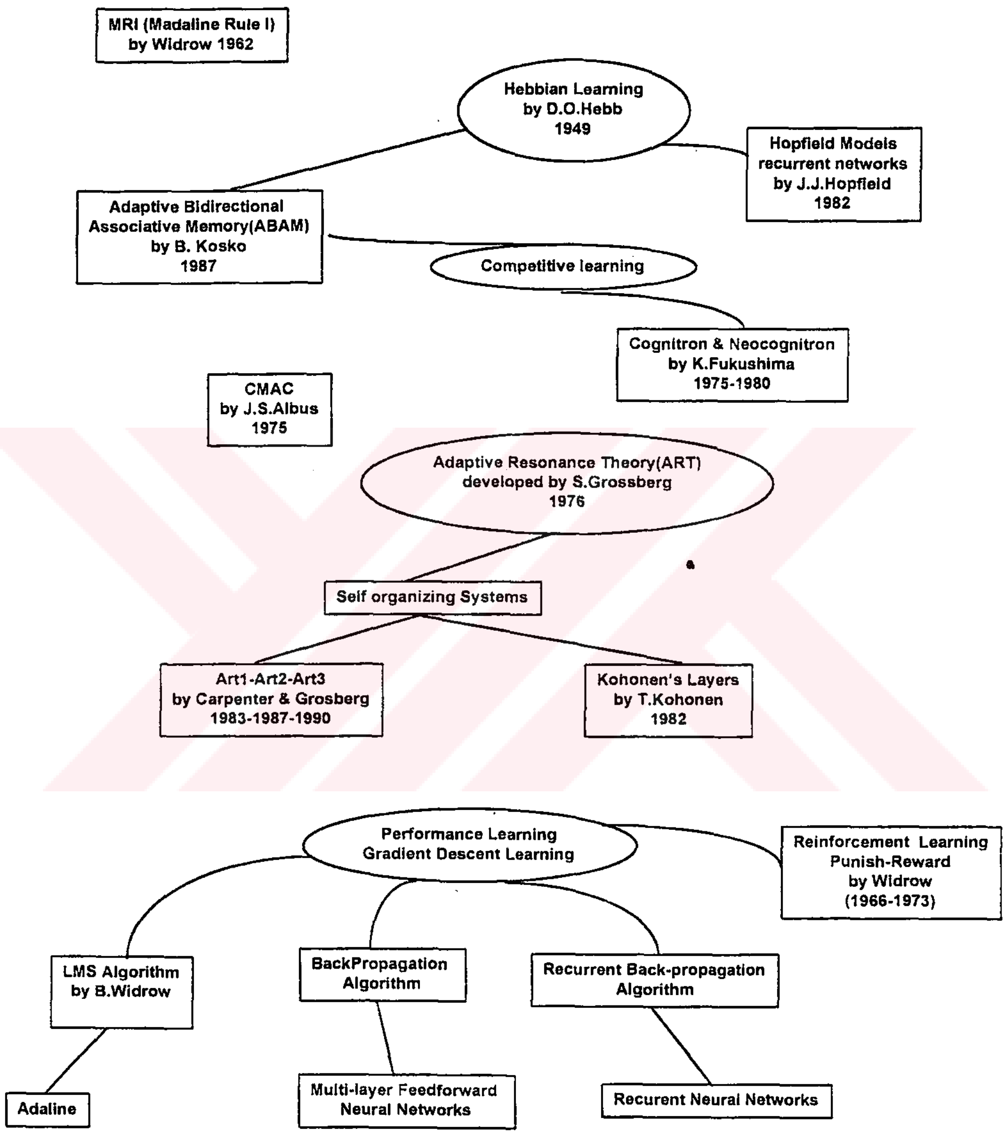
## 6.1 Artificial Neural Networks

Neural networks are embeded in control theory and adaptive signal processing as an inspiration from neuroscience that investigates the structural analysis of the brain in the sense of biology. Functional capacity of brain and its complexity have always been attractive for researchers. Anyone can easily notice the superiority of the brain to digital computers. As our engineers, we have endeavored to build the new machines with the help of research results articulating the structural and working principles of biological nervous system of the brain. Because brain behavior has some recognizable superiority in the functional comparison with conventional computation machines: it is massively parallel, robust and fault tolerant, flexible (learning ability), very fast to retrieve the information (recognizing speech and images). As a result of neuroscience, neurons refer to processing elements and synapses are connections in computation science. Great number of neurons that each process the simple functions are connected to provide collective behavior under the massive parrallesizm. The collective behavior of neurons (simple basic processing units) can solve very complicated numerical problems that have not been handled with the conventional computation methods. This is the main fascinating property of artificial neural networks (networks constructed nonbiologicaly). They can be used to grasp very complicated functions without being an expert. Needless to say, this property of neural networks (learning ability of the nonlinear dynamics) may be utilized in dynamic system control for highly nonlinear plants or unknown models. So far, all the conventional methods including the optimal control needed very complicated computational tools to control the dynamics. Fuzzy logic control is one alternative way to avoid these computation intensive methods. Again it is inspired from human-like thinking in decision making as a consequent of human behavioral analysis. Neural networks appear as another research topic in engineering inspired from brain structure. This research may be backdated to 1940's and several important contributions have been carried out up to 1969 [30][32]. In this year, book 'Perceptrons' [31] published abouth the restrictions of perceptrons proposed by Rosenblatt, F.[30], caused the decrease in the popularity of field. Until the recent developements have solved the difficuilties arisen in perceptrons in 1985 [33], a few researchers stayed in this field[34]. These important developments are the feedforward networks and back-propagation algorithms.

## 6.2. Neural Networks and Learning Algorithms

Madaline I architecture of B.Widrow is recognized as one of the early works in this field [34]. Hebbian learning proposed by Hebb in 1949 may be be classified in the coincidence learning as a basic learning law (p. 50 in [27]). Hopfield developed a new model using Hebbian learning in 1982. The other important learning types are competitive learning and performance learning. Filter learning can be added to this list [27]. The cognitron and neocognitron are the examples of competitive learning. Both of them were developed by K.Fukushimo in 1975 and 1980 [35][36]. The other competitive learning examples; ART1, ART2, ART3[40] were proposed using the adaptive resonance theory (ART) developed by S.Grossberg[39]. Kohonen's layer[41] is another recognizable model can be counted in the self organized structures with ART1, ART2 and ART3. Kosko's adaptive bidirectional associative memory (ABAM) model is a combination of Hebbian and competitive learning[25]. Performance learning is the most common learning type used in numerous studies.Most of the performace learning architectures are based on gradient descent techniquest. Adaptive linear combiner (Adaline) uses least mean square (LMS) algorithm developed by B.Widrow[26]. Multi layer feedforward networks have been trained by back-propagation algorithm derived by using gradient descent. Fig 6.2.1 shows the neural networks and learning algorithms stated in this section.

Learning algorithms are catogorized in two groups. There are supervised and unsupervised learning. In supervised learning, network is trained with the help of a



MRI (Madaline Rule I) by Widrow 1962
Hebbian Learning by D.O.Hebb 1949
Hopfield Models recurrent networks by J.J.Hopfield 1982
Adaptive Bidirectional Associative Memory(ABAM) by B. Kosko 1987
Competitive learning
Cognitron & Neocognitron by K.Fukushima 1975-1980
CMAC by J.S.Albus 1975
Adaptive Resonance Theory(ART) developed by S.Grossberg 1976
Self organizing Systems
Art1-Art2-Art3 by Carpenter & Grosberg 1983-1987-1990
Kohonen's Layers by T.Kohonen 1982
Performance Learning Gradient Descent Learning
Reinforcement Learning Punish-Reward by Widrow (1966-1973)
LMS Algorithm by B.Widrow
BackPropagation Algorithm
Recurrent Back-propagation Algorithm
Adaline
Multi-layer Feedforward Neural Networks
Recurent Neural Networks

**Fig. 6.2.1 Neural Networks and Learning Algorithms**

supervisory signal. Most of neural network applications related with control are the examples of supervised learning. An expception is the temporal back-propagation algorithm developed to train the fuzzy neural networks [42]. Unsupervised learning in contrast to supervised learning, uses only the input paterns and classifies them. Unsupervised learning is mainly used in recognition (Vector Quantizations are typical examples).

## 6.3. Neural Network Structures

Neural networks are said to be linear unit or nonlinear unit, single-layer or multi-layered, feedforward or nonfeedforward (recurrent). All of them defines the structure of neural network.

Fig 6.3.1 illustrates the simple processing unit called neuron in feedforward neural network(FNN).

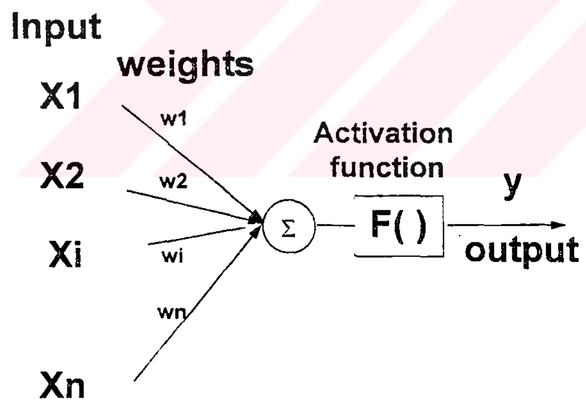


Fig 6.3.1 Neuron:A Processing unit

Function F(.) determines the linearity of network. In linear neural networks F(.) is a linear function such as Widrow's Adaline [26]. Function F(.) is also called activation output of neuron. It is generally choseen as a bounded and monotonically increasing function. This property is required for the neural network's stability. 'Back propagation in non feed-forward networks' in page 74 of [43] investigates the required stability conditions in the sense of Lyapunov's stability theory given in 3.4.1.

Fig 6.3.2 shows the structure of the feedforward multi-layered neural network. the first layer is input layer. The layers between the input and output layers are called hidden layers. In a FNN model every layer can include neurons (proceessing element) in the desired number.

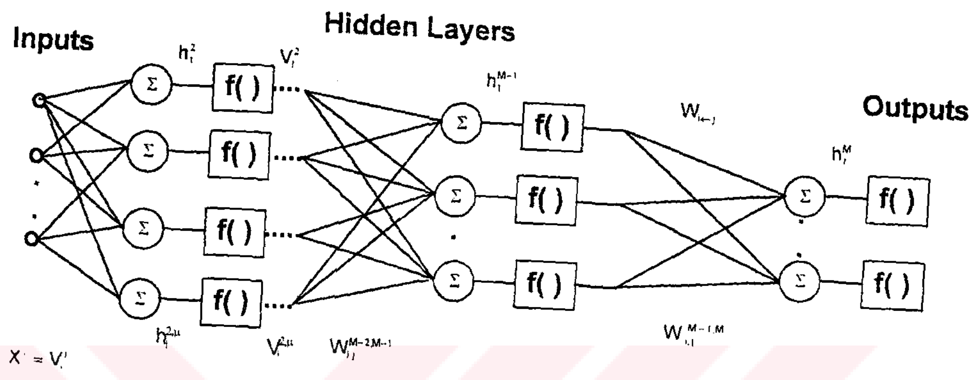


Fig. 6.3.2 Feedforward multi-layered neural network

In recurrent neural networks (RNN), present and previous outputs of neurons are fedback to layers. RNNs denote dynamic behavior [73], since they have stored the data. Because of their dynamic learning behavior. RNNs have been used frequently in the dynamic control and system identification [72][74].

## 6.4. Adaptation of Neural Networks

According to reference [28], adaptation is possible in three levels. These are function level adaptation, parameter level adaptation. structure level adaptation.

Function Level Adaptation: Activation function is changed according to input signals.

Parameter Level Adaptation: Unknown parameters are identified to satisfy relationship between inputs and outputs. Parameter level adaptation is related with parameter estimation (discussed in chapter four) and identification problem.

Structure Level Adaptation: Structure of neural network is adapted such as neuron numbers in layers and layer numbers.

In this chapter, we will use the fixed activation functions and structures. In [28] a special Fluctuated Distortion Measure is proposed to be used in an adaptable structure feedforward layered neural networks, FUNNET(FUNction NETwork).

In later sections, most common algorithm for FNN, back-propagation will be presented. Also extended Kalman filter will be used to estimate the weights of a FNN.

## 6.5. Parameter Level Adaptation

In this section, we introduce the learning algorithms for neural networks. The adaptation rules for the weights of a neural networks are called the learning algorithms. The optimal control and adaptive control have studied this problem as a parameter identification or estimation problem for the last 30 years. Essentially, algorithms developed in adaptive signal processing have been applied to neural networks to identify the weights properly. For example, extended Kalman filters have been used in neural networks as a nonlinear minimum variance estimator[77]. Hence, for the neural networks, we will modify the results of optimal control and estimation theory which are presented in the chapters three and four.

### 6.5.1 Backpropagation (BP) Algorithm for FNN

The backpropagation algorithm is based on gradient descent to minimize a defined performance. The error performance is defined by

$J(w) = \frac{1}{2}\sum_{\mu}\sum_{i}(O_i^d - O_i^M)^2$ where $O_i^d$ is the desired output for the ith output.

for output layer

$O_i^M=f(h_i^M)=f(\sum W_{ij}^{M-1,M}V_j^{M-1})$

Activation function f(x) is

$$f(x)=\frac{1}{1+e^{-x}}$$

The weighting matrix W is updated according to gradient descent.

$$W_{ij}^{M-1,M}(k+1)=W_{ij}^{M-1,M}(k)+\Delta W_{ij}^{M-1,M}$$
$$\Delta W_{ij}^{M-1,M}=-\alpha\nabla_w J(W)=-\alpha\frac{\partial J}{\partial W_{ij}^{M-1,M}}$$

where $\alpha$ is the learning rate $0<\alpha<1$.

For the convergence and derivation of B.P. refer to page 115 of [29].

### 6.5.2 Least Mean Square (LMS) Algorithm

LMS was developed by B.Widrow for adaline in adaptive signal processing(p. 99 of [26]). Main objective of LMS algorithm is to minimize the following performance index using gradient descent method:

$$J(W_k)=E[\varepsilon_k^2]$$

$\varepsilon_k$ is error defined by $\varepsilon_k=d_k-X_kW_k$ for the kth sample. In adaline, (adaptive linear combiner) the activation function is unitary and there exists only one layer with weights $W_k$ [34]. $d_k$ represents the desired output. LMS algorithm minimize the mean square error (MSE) surface.

### 6.5.3 Learning Algorithm via Extended Kalman Filter

Kalman filter, studied extensively in chapter four can be used in nonlinear estimation. For this application Kalman filters must be modified. This modified Kalman filters are called extended Kalman filter[13] uses linearized model of nonlinear system. Kalman filter is used in neural networks to identify the weights as a nonlinear minimum variance estimator. Extended Kalman filter application to neural networks increase the learning rate apparently with respect to B.P. algorithm.

### 6.5.4 Other Algorithms and Improving Learning

Some other methods may be used to adapt the weights in neural networks. Newton's method uses Hessian matrix of performance index. The steepest descent method is the simplest approach using line search. However, backpropagation algorithm presented in section 6.5.1 may failure to find the global minimum due to local minima. To overcome this problem, a momentum term is added in updating equation:

$$\Delta W_{ij}^{M-1,M}(t+1) = -\alpha \nabla_w J(W) + \eta \Delta W_{ij}^{M-1,M}(t) = -\alpha \frac{\partial J}{\partial W_{ij}^{M-1,M}} + \eta \Delta W_{ij}^{M-1,M}(t)$$

where $\eta$ is the momentum parameter between 0 and 1.

Similarly, fast learning algorithms are proposed for neural networks in literature[76].Hence far, all the learning algorithms including extended Kalman filter methods are linear and quadratic programming problems. We must be familiar with these terms from optimal control studied in the previous section that algorithms are linear and quadratic because weights are updated using linear rules to minimize the performance index defined in the quadratic form. Hence, the global stability of neural networks which are nonlinear models, becomes important for the convergence speed of a specific algorithm[78]. In contrast to linear quadratic programming of neural networks, Lagrange multipliers can be used in the learning as a global optimization tool[75].

### 6.7. Important Theorems in Neural Networks

For the performance guarantee of neural networks, Kolmogorov and The Stone-Weierstrass theorems are important.

### 6.7.1 Kalmogorov's Theorem and Its Application to Neural Networks

One of most influential and surprising theorem in neural networks is Kolmogorov's theorem. This famous theorem has been conceived to solve the 13th problem of Hilbert and utilized in neural networks. Briefly, this theorem proves that neural networks with at least one hidden layer can learn any continuous nonlinear function (p. 122 of [27]).

### 6.7.2. The Stone-Weierstrass Theorem

Another powerful theorem to examine the nonlinear mapping ability of a network is the Stone-Weierstrass Theorem[44]. From the application of theorem, it is guaranteed that a neural network biased unitarily can map any nonlinear continuos function in the unit hypercube.

## 6.8. Neural Networks in System Identification and Control

Applications of neural networks spread in a broad spectrum. They have been used successfully in a wide variety field such as modeling chemical process[71], heat transfer data analysis[70], automatic braking control systems[68], solving linear and differential equations[67][69].

Neural networks have been applied to control systems in numerous examples because of their capability of mapping nonlinear functions. Theoretical background of neural networks in control and system identification has been studied extensively[55][56][57][58][59][60][61]. Neural networks can be used as a nonlinear estimator. They can also be used as a feedforward controller like cerebellar model articulation controller(CMAC)[38] . CMAC is a feedforward controller uses a look-up table. In CMAC example a neural network can learn a control surface produced by a reference controller.

Neural networks can find application field in the adaptive control[80] strategies. Two adaptive control structures are Model Reference Adaptive Control

[81] and Self-Tuning Regulator(STR). Fig 6.8.1 illustrates the block diagram of MRAC. MRAC is also called direct adaptive control since any error between plant and reference model is used directly for the adaptation of control input to compensate the error. STR is indirect adaptive control; a parameter identification is used to obtain the unknown parameters of plant's model. This identified parameters are used to redesign a controller. Fig 6.8.2 illustrates the block diagram of STR. The objective of adaptive control is to find the control rules for slowly time-varying plants.

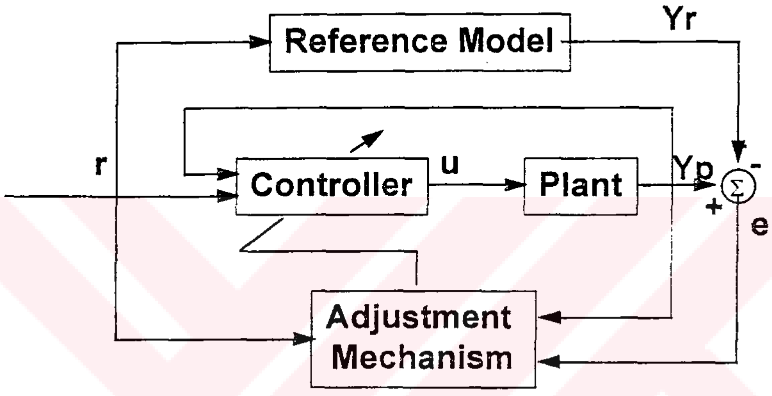


Fig 6.8.1 Model Reference Adaptive Control (Direct Adaptive System)

Neural networks can be used in the direct and indirect adaptive control of nonlinear plants[56]. In MRAC (direct adaptive control), nonlinear control laws are updated taking account the error between reference model and plant. These nonlinear adaptation laws(adaptation mechanism in fig 6.8.1) are derived to minimize a performance index using several methods such as gradient descent. For the stability criteria, some adaptation rules are determined under the Lyapunov stability theory. Some others are derived using hyperstability theory. Consequently, these adaptation rules may be very computation-intensive and complicated. K.S. Narendra proposed neural network architectures for adaptive control in [56]. Fig 6.8.3 shows the direct adaptive control of nonlinear plant using the neural networks.

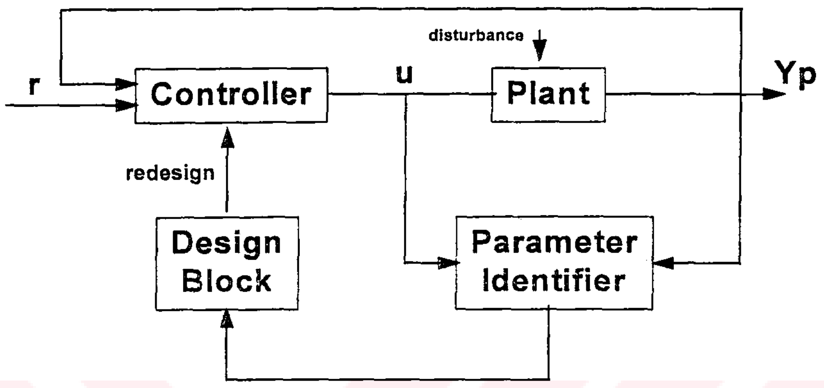


Fig 6.8.2 Self-Tuning Controller(Indirect Adaptive System)

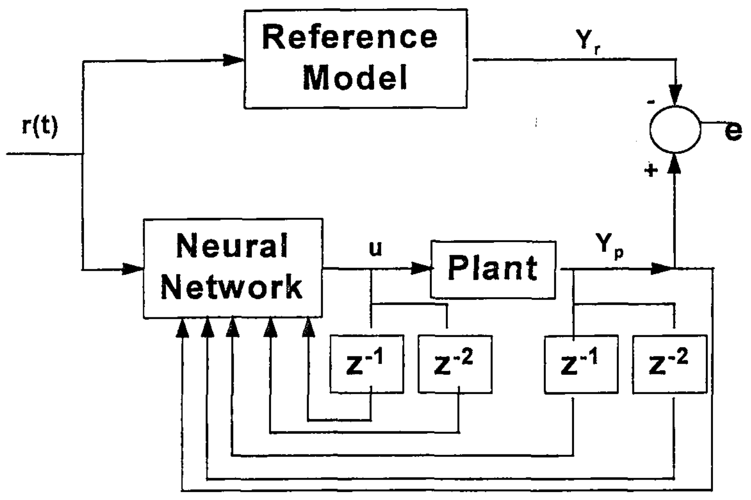


Fig 6.8.3 Direct adaptive control of nonlinear plants using neural networks

Neural networks can be used as direct transfer function identifier (fig 6.8.4-a) or inverse transfer function identifier(fig 6.8.4-b)[57] for nonlinear plants. Similarly, neural networks can be used as a nonlinear estimator in fig 6.8.4-a.

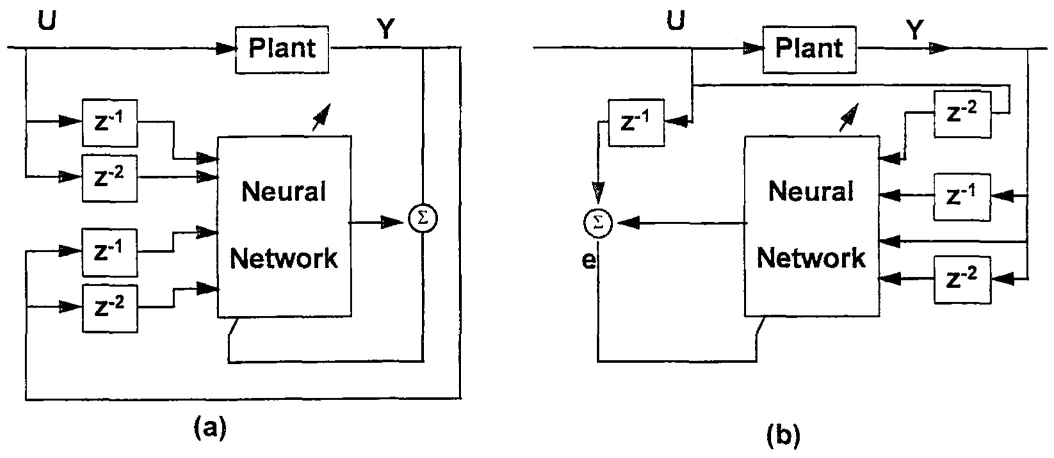


Fig 6.8.4 a)Block diagram of direct transfer function identifier
b)Block diagram of inverse transfer function identifier

## 6.9. Neurocontrol of Nuclear Power Plants

Recently, neural networks have been applied in power prediction[63], signal prediction [64], alarm processing and diagnostics [65] and control [62][66] of nuclear power plants. We will present the results of a direct control of a nuclear power plant using feedforward neural networks (FNN). To examine the flexibility and nonlinear mapping neural networks and compare the neurocontrol with FLC for nuclera plants, we have prefered to use FNN in the control system configuration ilustrated in Fig.5.3.2. A FNN with 2 inputs,1 output and 2 hidden layers, has been trained to learn the same supervisory signals which were used to tune the fuzzy rules of a FLC in chapter five. FNN with 10 neurons in each hidden layer has been trained via extended Kalman filter learning algorithm. Error and change in error have been entered to the inputs of FNN. The single output of FNN has been used as a control input to plant. The code MLNNKF.m using extended Kalman filter learning for a FNN defined in vector M, has been run for the inputs error E and change in error DE and output $Y_c$ of LQG as a reference controller.Fig.6.9.1 shows the learning mechanism to train a FNN for the control surface produced by LQG. M is a vector defining FNN in MLNNKF.m. In our case, M is $[2\ 10\ 10\ 1]^T$ which refers to a FNN with 2 inputs, 1 output and 2 hidden layers which each has 10 neurons. From Fig.6.3.2, there are three weighting matrices for 2 hidden layers. Weighting matrices $W_{i,j}^{1,2}$, $W_{i,j}^{2,3}$, $W_{i,j}^{3,4}$ connect the inputs to the first hidden layer, the first hidden layer to the second hidden layer and the second hidden layer to output layer, respectively. The values of weighting matrices after 10th training

$W_{2X10}^{1,2}$ =

```
2.2940   0.3516
1.6532   0.6942
0.2782   0.8851
0.3467   0.3895
-2.6817   0.1351
-1.2998  -0.8039
-0.3864  -0.0566
3.9285  -0.3069
```

| | |
|---|---|
| -0.9750 | 0.3401 |
| -0.2326 | 0.9209 |

$W_{10X10}^{2,3}=$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0.7154 | 1.5669 | 0.5983 | -0.5824 | -1.8530 | 0.2577 | -0.3585 | 0.4887 | 0.2060 | -0.5887 |
| -0.4620 | 0.0502 | 0.3020 | 0.3838 | -0.5241 | 0.6195 | -0.0892 | 1.2684 | -0.8991 | 1.0081 |
| -0.9333 | -1.2658 | 0.1587 | -1.0419 | -0.4629 | -0.0186 | 0.2884 | 0.7380 | -0.8178 | -0.1621 |
| -0.8416 | 1.0887 | 0.0136 | -0.2321 | -1.9234 | 0.3936 | -0.4659 | -0.1152 | -1.0024 | 1.0470 |
| -0.7661 | -0.0779 | 1.1135 | 0.7345 | 0.7194 | -0.6507 | -0.5922 | 1.4676 | -0.2465 | -1.3811 |
| 0.2331 | -0.1406 | -0.7701 | 0.2634 | 0.3770 | -0.1779 | 0.4006 | 0.5800 | 0.2147 | -0.9923 |
| 0.1576 | 0.8369 | 0.6477 | 0.2480 | -0.8675 | -1.1298 | -0.0427 | 0.6565 | -1.2967 | -1.6637 |
| 0.1655 | -0.9246 | 0.4714 | 0.1808 | 1.3725 | -2.2065 | -1.0428 | 0.7926 | 0.7593 | 0.9927 |
| -2.2217 | 0.4201 | -0.6851 | -3.0953 | 0.1900 | 0.2507 | -1.5502 | -1.2700 | 0.5792 | -0.2141 |
| 1.4340 | -0.9540 | 1.9924 | 0.7039 | 0.4904 | -1.6644 | 0.0250 | 0.2035 | 0.2443 | -1.7954 |

$W_{10X1}^{3,4}=[-1.9922 \;\; 0.4864 \;\; 0.7397 \;\; 0.1677 \;\; -0.5333 \;\; -0.4307 \;\; -0.8834 \;\; 0.8434 \;\; -0.9475 \;\; -0.5561]$

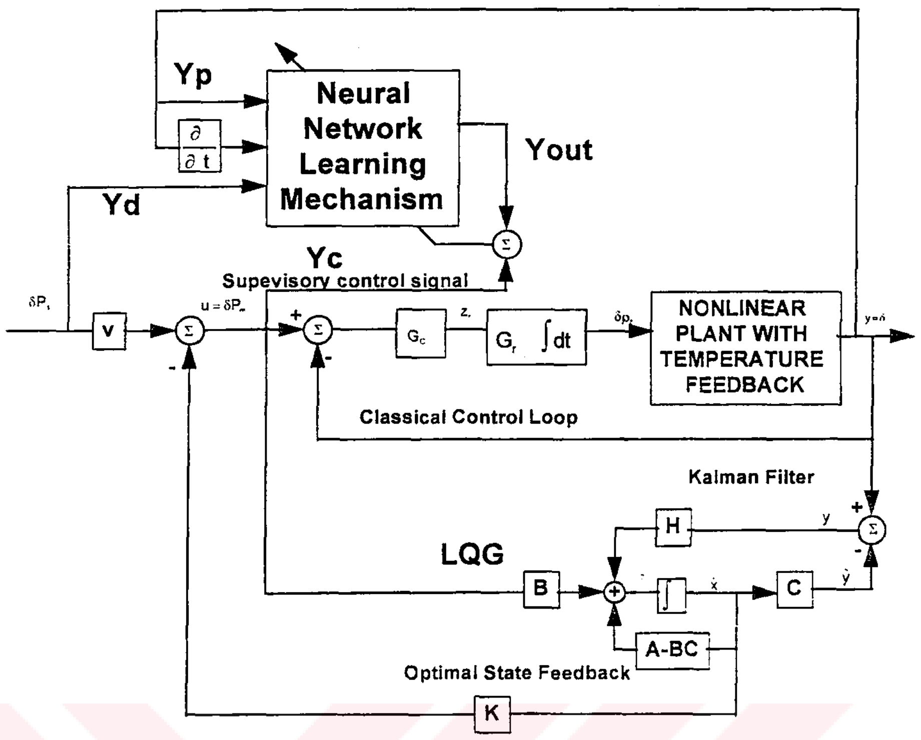
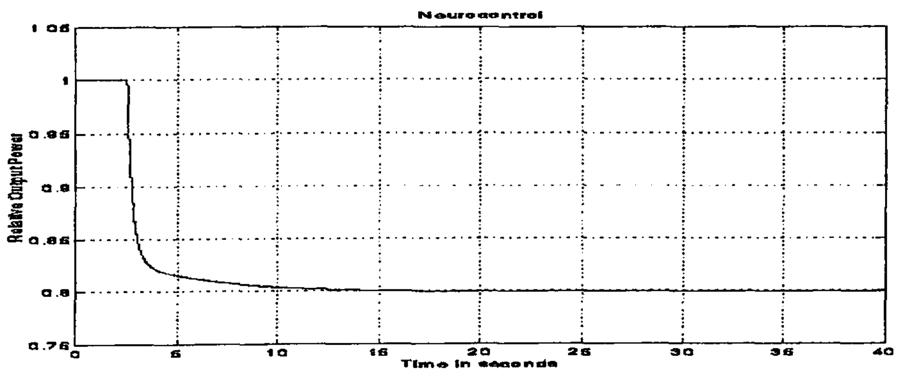
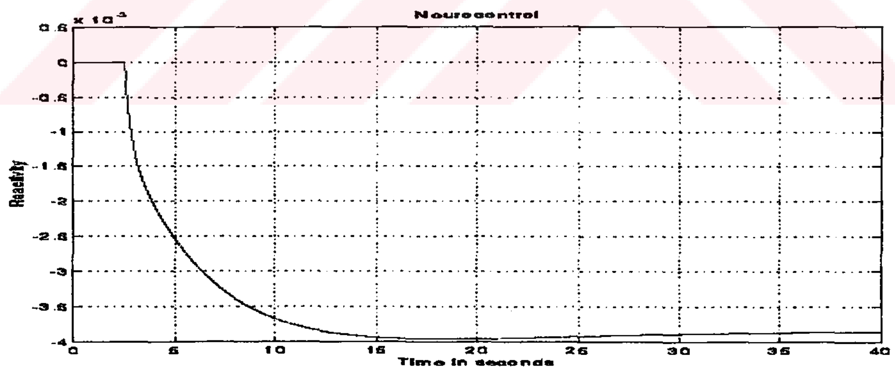


Fig.6.9.1 Neural network learning mechanism for optimal control surface to be used like CMAC

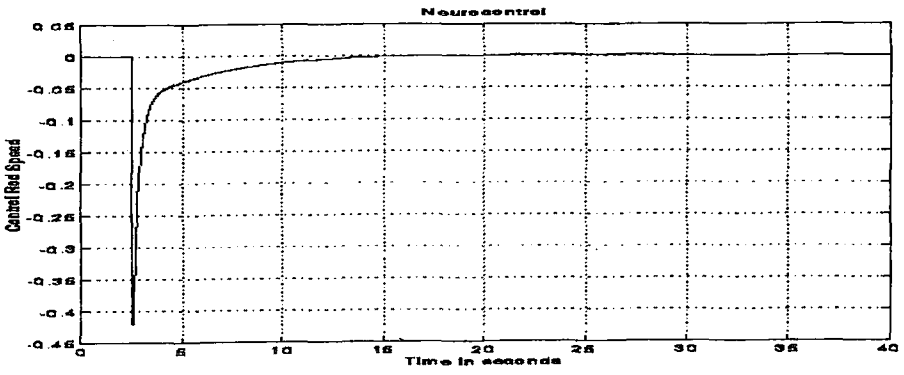
Nc_1.m has been run for the simulation of neurocontrol of the nuclear power plant. Fig.6.9.2 and Fig 6.9.3 ilustrate the results for 20% decrease in demand power($P_d$=0.8) and 10% increase in demand power ($P_d$=1.1).


Neurocontrol
Relative Output Power
Time in seconds
1.05
1
0.95
0.9
0.85
0.8
0.75
0
5
10
15
20
25
30
35
40

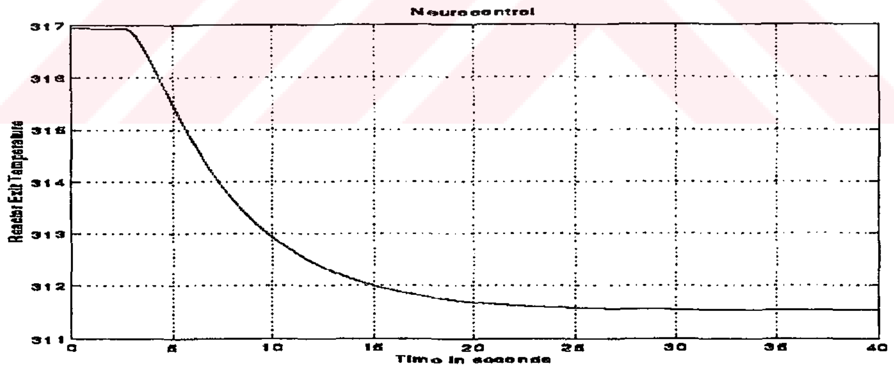
(a)


Neurocontrol
x 10⁻³
Reactivity
Time in seconds
0.5
0
-0.5
-1
-1.5
-2
-2.5
-3
-3.5
-4
0
5
10
15
20
25
30
35
40

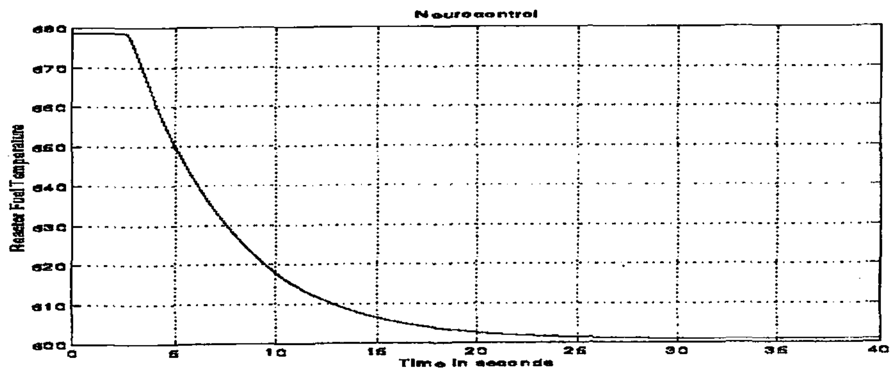
(b)


Neurocontrol
Control Rod Speed
Time in seconds
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
-0.3
-0.35
-0.4
-0.45
0
5
10
15
20
25
30
35
40

(c)

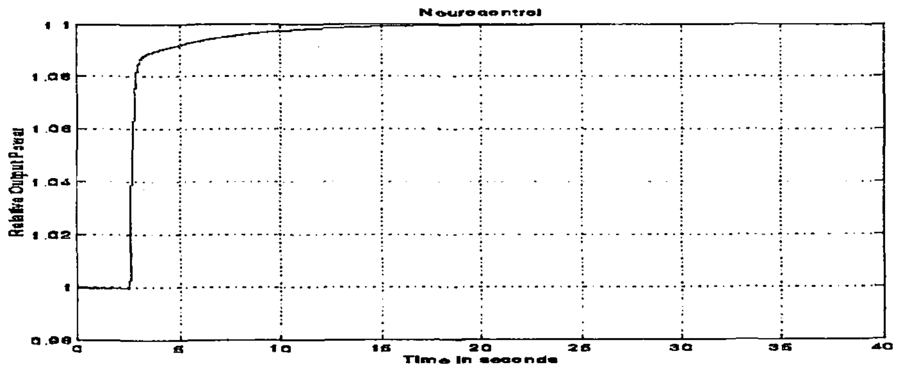

Neurocontrol
Reactor Exit Temperature
Time in seconds
317
316
315
314
313
312
311
0
5
10
15
20
25
30
35
40

(d)

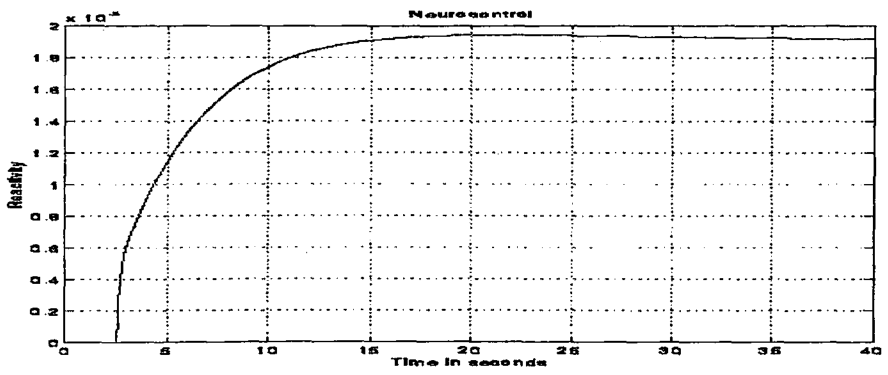


(e)

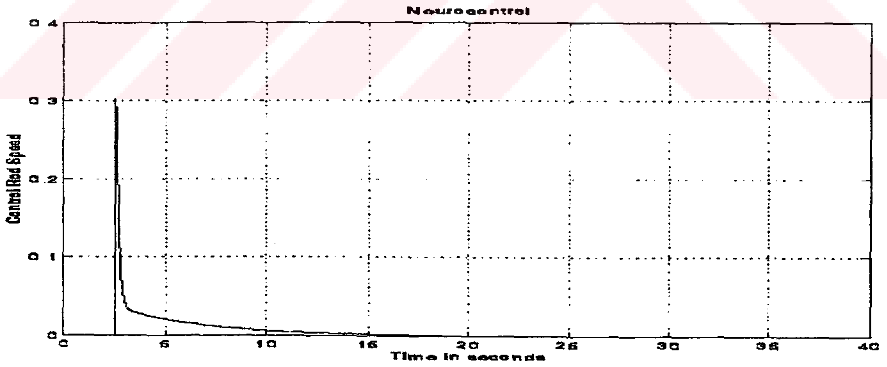
Fig.6.9.2 The simulation results of the neurocontrol of the nuclear power plant for 20% decrease in demand power ($P_d$=0.8) a) relative output power of plant b)reactivity c)control rod speed d)reactor exit temperature e)reactor fuel temperature.



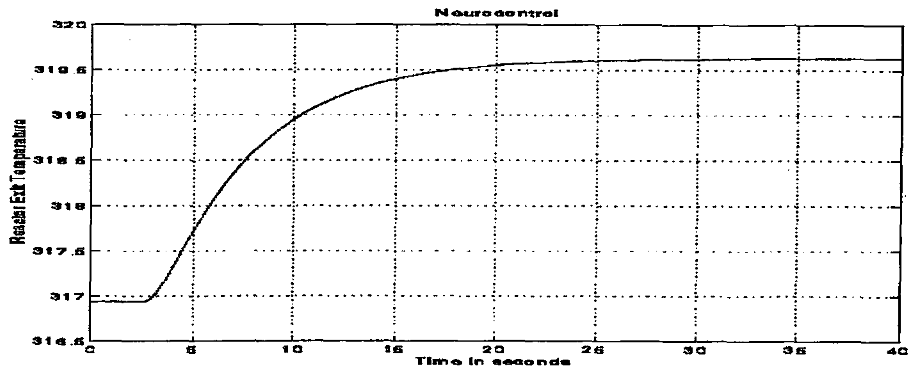
(a)


Neurocontrol
x 10⁻³
Reactivity
Time in seconds
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
5
10
15
20
25
30
35
40

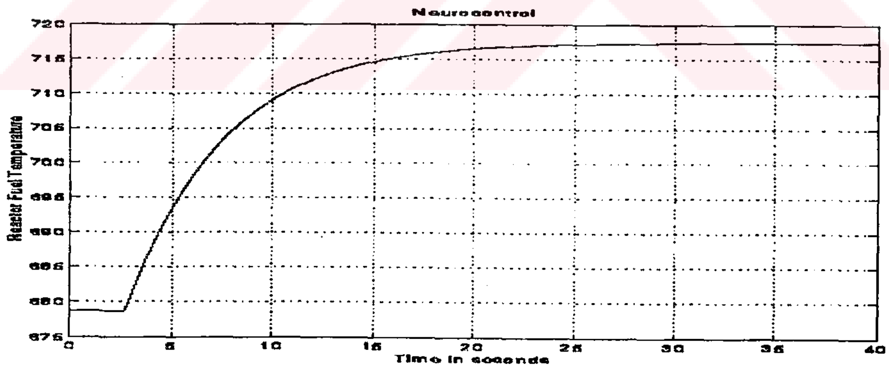
(b)


Neurocontrol
Control Rod Speed
Time in seconds
0
0.1
0.2
0.3
0.4
5
10
15
20
25
30
35
40

(c)



(d)



(e)

Fig.6.9.3 The simulation results of the neurocontrol of the nuclear power plant for 10% increase in demand power ($P_d$=1.1) a) relative output power of plant b)reactivity c)control rod speed d)reactor exit temperature e)reactor fuel temperature.

## 6.10. Conclusion

In our application, a FNN has been trained for a control surface produced by LQG as a reference controller to construct a neurocontroller for nuclear power plants in design stage. Designed neurocontroller modified demand power signal using the dynamic output of the plant. Decision of neurocontroller is based on experience taught by supervisory controller but not a mathematical model. It is seen from the simulation results in Fig.6.9.2 and Fig.6.9.3 that action of neurocontroller is very similar to the action of optimal controller. We can conclude from the results of our application that a neurocontroller can behave like an optimal controller and produce the similar control inputs. It is an advantage of neurocontroller that it can make decisions only using the output of a plant without needing state variables which are required for LGQ. The final remark is for the learning capability of neural networks: although neurocontroller has been trained for a control input produced by the LQG to reduce the output power to its %60 ($P_d$). neurocontroller achieved a similar performace for a %10 increase in the demand power($P_d$).

CHAPTER SEVEN

# CONCLUSIONS AND DISCUSSIONS

In this work, we have designed several controllers for nuclear power plants. These are classical output controller(COC), state feedback controller with state estimation (SFCSE) using pole placement techniques, deterministic optimal state feedback controller (LQ), stochastic optimal controller (LQG), automatically tuned fuzzy logic controller(FLC) and neurocontroller.

## 7.1 Summary of Work

In order to define the required qualifications sought for a controller of nuclear plants. we have investigated the dynamical behavior of plants presenting a nonlinear model of the nuclear plant in Chapter Two. In the same chapter, classical output control (COC) and state feedback control with state estimation (SFCSE) have been simulated for the derived nonlinear model of the nuclear plant. SFSE has been designed using the linearized model of plant. It is seen from the results of the Chapter One that reactivity denotes the most nonlinear characteristics among the other state variables as it is expected from the nonlinear model. It is concluded from the dominant eigenvalues of linearized model at different power levels that major nonlinearity source is mostly due to temperature feedback. In SFSE, we have used Ackermann's formulas to place the poles for the desired response of plant. We have also defined the stiffness which results in some numerical difficulties.

In Chapter Three, we have introduced the concept of optimality and designed optimal state feedback controller for nuclear plants. To achieve this goal, we have solved algebraic Riccati equation(ARE) arising in optimal control. We have concentrated on the accuracy in the solution of ARE. We have showed in a numerical example that stiffness arising in the model of plant, results in the inaccurate solutions for some algorithms solving the ARE.

In Chapter Four, It is seen from the simulation results that estimator performance has been improved in Kalman filter.This also improves the robustness of controller.

In Chapter Five, an automatically tuned fuzzy logic controller has been designed and simulated for the nonlinear model of nuclear plant. Fuzzy rules has been tuned via least square estimation using a LQG as a reference controller.

In the final chapter, we have introduced neural networks and used them in the control of nuclear plants. A feedforward neural network has learned the control surface produced by the same LQG. FNN has been trained using learning via the extended Kalman filter.

## 7.2. Comparison of Controllers

We will compare the controllers in the sense of intelligent control , robustness and structure.

### 7.2.1 Structural Comparison

All the controllers designed in this work can be incorporated in mainly two groups, servocompensators and state feedback controllers(regulators). Servocompensators includes output unitary feedback and feedforward compensators. COC, FLC and neurocontrol (in our application) are servocompensator examples. In contrast to state-feedback controllers, they do not require the states of plant. Control input is a function of error between output of plant and demand signal and its derivative(change in error). COC includes only static compensator, while FLC and neurocontrol use both static and dynamical compensators. We can count proportional derivative (PD) and proportional integral (PI) controllers from conventional control in this group. COC, PI and PD controllers are linear controllers. Since FLC and neurocontrol produce the their control outputs using complex nonlinear functions, they are nonlinear controllers.

### 7.2.2 Intelligent Control

Intelligent control includes all the methodologies to combine all the conventional control problems to solve new challenging problems. Conventional control problems are solved in the lower level of intelligent control. FLC and neurocontrol are intelligent control methodologies. In FLC, output of FLC is a combination of several PD controllers. FLC supervises these PD controllers according to the fuzzy rules and information-based membership function. It is appropriate example for the definition of intelligent control that a conventional method least square estimation is used in the tuning mechanism of FLC. Similarly, neural network is trained using the extended Kalman filter. It is seen from neurocontrol application in chapter six that neural networks can learn complex control surfaces easily. Several control surfaces produced by different conventional controllers can be combined in a neurocontroller. FLC uses the linguistic knowledge's about the plant with no need of mathematical model. In our work, automatically tuned FLC and neurocontroller have learned the control surface produced by the LQG as a reference controller.

### 7.2.3 Robustness

The robustness is another criteria to test the stability, steady state error of output and dynamical performance of close loop system under a given class of variations of the open-loop dynamics. We can conclude from the simulation results that LGQ is the most robust controller which can compensate for the variations. Kalman filter in LQG has been designed taking some inaccuracies into account using stochastic model. We can improve the robustness of LQG controller design by linearizing the model equations at different power levels. A neurocontroller trained for the LQG designed at the different power levels provides a more robust controller.

## 7.3 Conclusions and Future Work

Initially, application of a broad spectrum of control design procedures, ranging from the most elementary classical output control to robust optimal control has been considered for a PWR core modeled with point kinetic equations and lumped thermal-hydraulic description. to demonstrate the benefits offered by the more advanced methods. The LQG controller was chosen as the reference model. Then two separate controllers, one based on fuzzy logic, and the other on artificial neural networks have been proposed and trained by feeding the output of the LQG controller as a supervisory signal.

FLC and neural networks can map nonlinear functions. In control applications they can learn control surfaces produced by a reference controllers. In this work we showed that fuzzy system and neural networks can be utilized in the control of nuclear plants. They can combine different reference controllers each of which is an expert for a different operating region of the plant, in a single controller with some structural advantages. This provides more intelligent and robust controllers for nuclear plants.

Suggestions for Future Work.

i-A more realistic and detailed models can be used to simulate the reactor core.
ii-The number of rules that comprise the data base of the FLC can be reduced
Presumably not more than 3 to 5 rules are needed.
iii-The inlet water temperature is assumed to be constant. However, we may incorporate the reactor core into the loop of a power plant whose control is in question.
iv-Other ANN architectures , for example recurrent networks may be implemented.
v- State identification may be done with the help of an ANN.
vi-Controller parameters obtained for a nonlinear system linearized over a wide range of power setpoints can be interpolated seamlessly with the help of an ANN.
vii. A 'Reactor Control Toolbox' for training students can be prepared. based on the results presented in this work.

## References

[1] Edwards R.M.,K.Y. Lee K.Y. and Schultz M.A.,State Feedback Assisted Classical Control: An Incremental Aproach to Control Modernization of Existing and Future Nuclear Reactors and Power Plants, Nuclear Technology,vol.92,nov.1990,167-185

[2] Bernard J.A.and Lanning D.D., Considerations in the design and Implementation of Laws for the Digital Operation of Research Reactors,Nuclear Science and Engineering:110,425-444(1992)

[3] Bernard J.,Digital Control of Nuclear Reactors,Trans. Am. Nucl. Soc.,64(1991)

[4] Phillips C.L.,and Harbor.C.L., Feedback Control Systems,Prentice Hall, NJ (1988)

[5] Edwards.R.M. Lee K.Y., and Ray A.,Robust Optimal Control of Nuclear Reactors and Power Plants',Nuclear Technology,vol 98,may.,1992

[6] Shankar.P.V.. Srikatiah G.and Pai.M.A.,Optimal Control of a Boiling Water Nuclear Reactor,Annals of Nuclear Energy, vol.3 pp 531-538 Pergamon Press.1976

[7] Fij X.and Fu L.. Optimal Control System Design of A Nuclear Reactor By Generalized Spectral Factorization', Int.J.Control, 1988, vol.47. no.5. pp.1479-1487

[8] Yim M.S.,Christenson J.M.. Application of Optimal Control Theory to a Load -Following Presssurized Water Reactor. Nuclear Technology vol.100.Dec.1992 361-376

[9] Anderson.B.D,Moore.J.B.,Optimal Control:Linear Quadratic Methods,Prentice-Hall Inc.,NJ 1989

[10] Bagchi A.,Optimal Control of Stochastic Systems,Prentice Hall,1993

[11] Bryson A.E.,Ho Y.C., Applied Optimal Control,Blaisdell Publishing Company,1969

[12] Lewis F.L.,Optimal Estimation,John Wiley & Sons Inc., 1986

[13] Gelb A, Applied Optimal Estimation,The M.I.T. Press,1974

[14] Ramaswamy P.,Edwards M.R., and Lee K.L., An Automatic Tuning Method of A Fuzzy Logic Controller for Nuclear Reactors,IEEE Trans.on Nuclear Science, vol.40, no.4, August 1993

[15] Raju G.V.,Zhou.J., Application of Fuzzy Rule Based Techniques to Controller Design of a Power Plant, The 8th Power Plant Dynamics, Control and Testing Symposium,vol.1.pp 27.01-11,Knoxville, May 27-29,1992

[16] Ikonomopoulos.A.,Tsoukalas.L.,and Uhrig.R.E,A Fuzzy-Neural Methodology For Power Plant Virtual Measurements, The 8th Power Plant Dynamics, Control and Testing Symposium, vol.II, pp 48.01-07, Knoxvile. May 27-29,1992

[17] Matsuoka H., A Simple Fuzzy Simulation Model For Nuclear Reactor System Dynamics. Nuclear Technology, vol.94, May 1991

[18] Pedrycz W., Fuzzy Control and Fuzzy Systems,Research Studies Press Ltd.,1993

[19] Sugeno M.,Industrial Applications of Fuzzy Control,Elsevier Science Publishing Company Inc.,1985

[20] Takagi T., Sugeno M., Fuzzy Identification of Systems and Its Applications to Modeling and Control, IEEE Trans. on Syst., Man, and Cyb., vol.SMC-15, no.1, pp.116-132 January/February 1985.

[21] Lee C.C.,Fuzzy Logic in Control Systems:Fuzzy Logic Controller-Part I,IEEE Trans.on Sys.Man and Cyb.,vol.20, no.2,March/April 1990 404-417

[22] Lee C.C.,Fuzzy Logic in Control Systems:Fuzzy Logic Controller-Part II IEEE Trans.on Sys.Man and Cyb.,vol.20, no.2,March/April 1990 419-435.

[23] Li Y.F., Lau C.C.,Development of Fuzzy Algorithms for Servo Systems,IEEE Control Systems Magazine,April 1989 65-71.

[24] Zadeh,L.A.,Fuzzy Sets,Information and Contr, Vol.8 pp338-353,1965.

[25] Kosko,B.,Neural Networks and Fuzzy Systems,Prentice-Hall Inc.,1992

[26] Bernard,W.,Stearns S.D.,Adaptive Signal Processing,Prentice-Hall Inc., !985.

[27] Hecht-Nielsen,R.,Neurocomputing,Addison-Wesley Company,1990.

[28] Lee,T.-C.,Structure Level Adaptation,Kluwer Academic Publisher.1991.

[29] Hertz,J.,Introduction to The Theory of Neural Networks,Addison-wesley Company,1991.

[30] Rosenblatt,F.,The Perceptron:A Probabilistic Model for Information Storage and Organization in The Brain,Pscychological Review, vol.65,pp386-408,1958.

[31] Minsky,M.,Perceptrons,Cambridge MA,MIT Press 1969.

[32] Widrow,B.,Adaptive Switching Cicuits, 1960 IRE WESCON Convention Record, New York, pp96-104,1960.

[33] Ackley,D.,A Learning Algorithm for Boltzman Machines,Cognitive Science,Vol.9.,pp.147-169 1985.

[34] Widrow,B.,30 Years of Adaptive Neural Networks:Perceptrons, Madaline, and Backpropagation, Proceed. of IEEE, Vol.78, No:9,September, 1990.

[35] Fukushimo,K.,Cognitron:A Self-organizing Multilayered Neural Networks,Biolog. Cybernetics,vol.20, pp 121-136,1975.

[36] Fukushimo,K.,Neocognitron: A Self-organizing neural network model for a mechanizm of pattern recognition unaffected by shift in position, Biolg.,Cybernetics,vol.36,pp193-202,1980.

[37] Widrow,B.,Punish/Reward:Learning with a Critic in Adaptive Threshold Systems, IEEE Trans.Sys.Man ,Cybernetics, vol.SMC-3 pp 455-465 September 1973.

[38] Albus,J.S.,A New Approach Manipulator control:Cerebellar Model Articulation Controller(CMAC),J Dyn.,Meas,Contr. vol 97,pp220-227,1975.

[39] Grossberg,S.,Adaptive Pattern Classification and Universal Recording II: Feedback, Expectation, Olfaction and Illustrations,Biolog,Cybernetics, vol.23, pp187-202,1976.

[40] Grossberg,S.,A Massive Parallel Architecture for a Self -Organizing Neural Patern Recognition Machine,Computer Vision, Graphics, and Image Processing,vol.37,pp.54-115,1983.

[41] Kohonen,T., Self-Organized Formation of Topologically Correct Feature Maps, Biolog,Cybernetics,vol.43,pp.59-69,1982.

[42] Jang J.-S., R., Self-Learning Fuzzy Controllers Based on Temporal Back Propagation, IEEE Trans. on Neural Networks. vol.3, no.5, pp.714-723, September 1992

[43] Aleksander,I.,Neural Computing Architectures:The Design of Brain-Like Machines,Nort Oxford Academic,1989.

[44] Cotter,E.N.,The Stone-Weierstrass Theorem and Its Application to Neural Networks,IEEE Trans.on Neural Networks,vol.1,no.4,December 1990.

[45] Kraft,G.L.,A Comparison Between CMAC Neural Network Control and Two Traditional Adaptive Control Systems,IEEE Cont.Sys.Mag.,pg 36-43,April 1990.

[46] Ben-Abdennour B., Edwards R.M., and Lee K.Y., LQG/LTR Robust Control of Nuclear Reactors with Improved Temperature Performance, IEEE Trans. on Nuclear Science, vol.39, no.6, pp. 2286-2294, December 1992.

[47] Ku C.C.,Lee Y., and Edwards R.M., Improved Nuclear Temperature Control Using Diagonal Recurent Neural Networks, IEEE Trans. on Nuclear Science, vol.39, no.6,pp. 2298-2308, December 1992.

[48] Choi C.C., Efficient Algorithms for Solving Stiff Matrix-Valued Riccati Differential Equations, PhD. Dissertation, ECE Dep., Univ. Calif., Santa Barbara, September 1988.

[49] Choi C.C., Efficient Matrix-Valued Algorithms for Solving Stiff Riccati Differential Equations, IEEE Trans. on Automatic Control , vol. 35, no.7, pp. 770-776, July 1990.

[50] Sagiroglu F.E., A Study on Efficient Algorithms for Solving Stiff Matrix-Valued Riccati Differential Equations, Graduate Project Report, Electrical and Electronics Engineering Dep., Hacettepe University, Beytepe, Ankara, June 1991.

[51] Gear C.W., The Automatic Integration of Stiff Ordinary Differential Equations, in Information Processing 68, A.J.H. Morell (Ed.) Nort-Holland, Amsterdam, 187-193, 1969.

[52] Brayton R.K., Gustyavson F.G., and Hachtel G.D., A New Efficient Algorithm for Solving Differential-Algebraic Systems Using Implicit Backward Differentiation Formulas, Proc. IEEE. 60. 98-108, 1972.

[53] Gear C.W., The Control of Parameters in The Automatic Integration of Ordinary Differential Equations, Internal Rept., Department of Computer Science, University of Illinois. 1968.

[54] Smith S.M., and Comer D.J., Automated Calibration of a Fuzzy logic Controller Using a Cell State Space Algorithm, IEEE Control Sytem Magazine. pp. 18-28. August 1991.

[55] Chu S.R., Shoureshi R., Tenorio M., Neural Networks for System Identification, IEEE Control Sytem Mag.. pp.31-34. April 1990.

[56] Narendra K.S., Parthasarathy K., Identification and Control of Dynamical Sytems Using Neural Networks. IEEE trans. on Neural Networks. vol.1.no.1. March 1990.

[57] Yamada T., Yabuta T., Dynamic Sytem Identification Using Neural Networks. IEEE Trans. on Sys. Man. and Cyb.. vol.23.,no.1, January/February 1993.

[58] Nyugen D.H. Widrow B., Neural Networks for Self-Learning Control Sytems. IEEE Control Sytems Mag., pp.18-23, April 1990.

[59] Guez A., Eilbert J.L., and Kam M., Neural Network Architecture for Control, IEEE Control Sytems Mag., pp.22-25 April 1988.

[60] Psaltis D., Sideris A., Yamamura A.A., A Multilayered Neural Network Controller. IEEE Control Sys. Mag.. pp.17-21, April 1988.

[61] Chen F.-C., Back-Propagation Neural Networks for Nonlinear Self-Tuning Adaptive Control, IEEE Control Sys. Mag., pp.44-48, April 1990.

[62] Ku C.C.,Lee Y., and Edwards R.M.,Neural Networks for Adapting Nuclear Power Plant Control for Wide-Range Operation, Trans. Amer. Nucl. Soc., vol.63, pp. 114-115, June 1991.

[63] Roh M.-S.,Cheon S.-W., Chang S.-H., Power Prediction in Nuclear Power Plants Using a Back-Propagation Learning Neural Networks. Nucl. Tech.,vol. 94, pp.270-278, May 1991.

[64] Kim W.J.. Chang S. H. Lee B.H., Application of Neural Networks to Signal Prediction in Nuclear Power Plant, IEEE Trans. on Nuclear Science, vol. 40, no.5, pp.1337-1341, October 1993.

[65] Cheon S.W., et.al., Application of neural Networks to multiple Alarm Processing and Diagnosis in Nuclear Power Plants, IEEE Trans. on nuclear Science, vol.40, no.1, Feb. 1993 pp.11-20.

[66] Ku C.C.,Lee Y., and Edwards R.M., Diagonal Recurrent Neural Networks for Nuclear Power Plant Control, The 8th Power Plant Dynamics. Control and Testing Symposium, vol. II, pp.61.01-14, Knowxville. May 27-29, 1992.

[67] Cichocki A., Unbehauen R., Neural Networks for Solving Systems of Linear Equations and Related Problems, IEEE Trans.on Circ. and Sys.-Part I, vol.39, no.2, February 1992.

[68] Ohno H.. et.al., Neural networks Control for Automatic Braking Control System. Neural Networks. vol. 7, no.8, pp.13031312. 1994.

[69] Lee H.. Kang I.S.. Neural Algorithm for Solving Differential Equations. Journal of Computational Physics 91. 110-131 (1990).

[70] Thibault J.. and Grandjean B.P.A.. A Neural Network Methodology for Heat Transfer Data Analysis. Int. J. Heat Mass Tran.. vol 34. pp.2063-2070. 1991.

[71] Bhat N.V..et.al., Modeling Chemical Process Systems via Neural Computaion. IEEE Control Systems Mag., pp.24-29. April 1990.

[72] You Y.. and Nikolaou M.. Dynamic Process Modeling with Recurent Neural Networks, AIChE Journal. vol.39. no.10. October 1993.

[73] Sato M.. A real Time Learning Algorithm for Recurrent Anolog Neural Networks, Biological Cybernetics, 62, 237-241 (1990).

[74] Parlos A.G., Chong K.T.. and Atiya A.F., Empirical Model Development and Validation with Dynamic Learning in the Recurrent Multilayer Perceptron, Nuclear Tech., vol 105. pp.271-290 Feb. 1994.

[75] Zhang S.. Constantinides A.G., Lagrange Programming Neural Networks, IEEE Trans. on Circ. and Syst.-II,vol.39. no.7, pp.441-452.

[76] Karayiannis N.B.. Venetsanopoulos A.N., Fast learning Algorithms for Neural Networks. IEEE Trans. on Circt. and Syst.-II, vol.39, no.7, pp.453-473, July 1992.

[77] Watanabe K., Fukuda T., Tzafestas S.G., Learning Algorithms of layered Neural Networks via Extended Kalman Filters, Int. J., Systems Sci.,vol.22.no.4, pp.753-768, 1992.

[78] Forti M., Tesi A.. New Conditions for Global Stability of Neural Networks with Application to Linear and Quadratic Programming Problems, IEEE Trans. on Circt. and Sys.-I, vol.42, no.7, pp.354-366 July 1995.

[79] Venugopal K.P. and Smith S.M., Improving the Dynamic Response of neural Network Controllers Using Velocity Reference Feedback, IEEE Trans. on Neural Networks, vol.4..no.2, pp.355-357, March 1993.

[80] Sastry S., Bodson M., Adaptive Control: Stability, Convergence, and Robustness. Prentice-Hall Inc., Englewood Cliffs, New Jersey 1989.

[81] Butler H.. Model Reference Adaptive Control. Prentice Hall International Ltd..1992.

[82] Walker V.A., The Future of Nuclear Power: One Perspective, IEEE Trans on Nuclear Science, vol.39, no.6., pp.2279-2281, December 1992.

[83] Hadley G., Linear Algebra, Addison-Wesley Publishing Company, Inc., Eighth Printing 1979.

[84] Laub A.J., A Schur Method for Solving Algebraic Riccati Equation, IEEE Trans. on Automatic Control, vol.AC-24, no.6, pp. 913-921, December 1979.

[85] Papoulis A., Probability, Random Variables, and Stochastic Processes, McGraw-Hill Book Company, 1984.

[86] Shanmugan K.S.. Breipohl A.M., Random Signals, John Wiley & Sons, Inc.,1988.

[87] Yager R.R., Modeling and Formulating Fuzzy Knowledge Bases Using Neural Networks. Neural Networks, vol.7. pp.1273-1283, 1994.

[88] J.Dombi, Membership Function as an Evaluation, Fuzzy Sets and Systems, vol.35, pp.1-21.1990.

[89] Golub G.H., Loan C.F., Matrix Computations, The John Hopkins University Press, 1989.

## APPENDIX A

## Simulation of Nonlinear Model for Nuclear Power Plants using BDFs

Nonlinear mathametical model arising in the modelling of nuclear power plants in the chapter two. is partitioned into linear differential equations. These linear differential equations are integrated constructing a Newton backward-difference interpolating polynomial. BDFs are succesfully applied in solving stiff differatial equations[51][52][48][49]. We have obtained fast and accurated simulation results of the plant using implicit backward differentiation formulas for the stiffness arising in the model of nuclear plant.

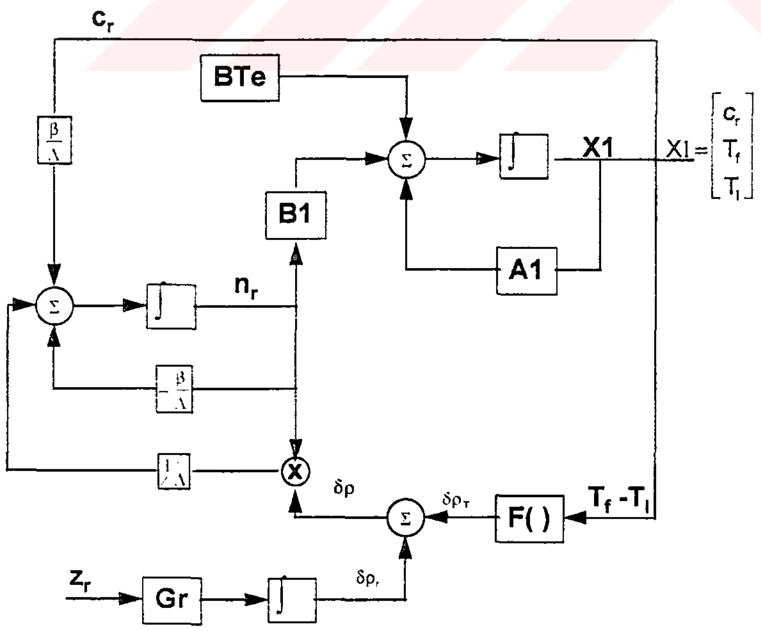


Fig App-A.1 Block Diagram of Simulation of Nonlinear Model

$$A1 = \begin{bmatrix} -\lambda & 0 & 0 \\ 0 & \frac{-\Omega}{\mu_f} & \frac{\Omega}{2\mu_f} \\ 0 & \frac{\Omega}{\mu_c} & \frac{-(\frac{\Omega}{2}+M)}{\mu_c} \end{bmatrix} \qquad 1 = \begin{bmatrix} \lambda \\ \frac{f_f P_{oa}}{\mu_f} \\ \frac{(1-f_f)P_{oa}}{\mu_c} \end{bmatrix} \qquad Te = \begin{bmatrix} 0 \\ \frac{\Omega}{2\mu_f}T_e \\ \frac{(M-\Omega/2)}{\mu_c}T_e \end{bmatrix}$$

$$\frac{\partial X1(t)}{\partial t} = A1 \cdot X1(t) + B1 \cdot n_r(t) + BTe \qquad \text{where } X1 = \begin{bmatrix} c_r(t) \\ T_f(t) \\ T_l(t) \end{bmatrix}$$

We obtain the following 5th order non-linear differential equation set that is based on point kinetics and a lumped parameter hydraulic model governing the dynamics of nuclear plant for one delayed neutron group normalized point kinetics equations are as follow:

$$\frac{dn_r}{dt} = \frac{\delta\rho - \beta}{\Lambda} n_r + \frac{\beta}{\Lambda} c_r \qquad (2.2.5)$$

$$\frac{dc_r}{dt} = \lambda n_r - \lambda c_r \qquad (2.2.6)$$

$$\frac{dT_f}{dt} = \frac{f_f P_o}{\mu_f} n_r - \frac{\Omega}{\mu_f} T_f + \frac{\Omega}{2\mu_f} T_l + \frac{\Omega}{2\mu_f} T_e \qquad (2.2.15)$$

$$\frac{dT_l}{dt} = \frac{(1-f_f)P_o}{\mu_c} n_r + \frac{\Omega}{\mu_c} - \frac{(2M-\Omega)}{2\mu_c} T_l + \frac{(2M-\Omega)}{2\mu_c} T_e \qquad (2.2.16)$$

$$\frac{d\delta\rho_r}{dt} = G_r z_r \qquad (2.2.17)$$

Total reactivity input to the points kinetics equations is the sum of control rod reactivity and temperature feedback reactivities of the fuel and coolant;

$$\delta\rho = \delta\rho_r + \alpha_f(T_f - T_{fo}) + \alpha_c(T_c - T_{co}) \qquad (2.2.12)$$

Block F( ) refers to the mathematical function of temperature feedback reactivity given in the equation(2.2.12).

INTEG6BD.m and INTEGBDF.m have been coded to integrate the linear differential equations in simulation block. Although parameters are time invariant in our model, in most general case models with time-varying parameters can be simulated easily. For example, LQG_2.m is a simulation program using the power dependent (time varying) parameters.

Functions INTEG6BD.m and INTEGBDF.m solve the following linear system;

$$\dot{X}(t)_m = B(t)_{mxm} X(t)_m + Q(t)_m$$

constructing a Newton backward-difference interpolating polynomial for the past values of X;

$$X_{k+1} = -\left\{\left[-\sum_{j=1}^{r}\frac{1}{j}\right] I_{mxm} + hB(t_{k+1})\right]^{-1}\left[\sum_{j=1}^{r}\frac{(-1)^{j-1}}{j}\begin{bmatrix} r \\ j \end{bmatrix} X_{k+1-j} + hQ(t_{k+1})\right]$$

where h is the step size and r is the order of backward difference equation,

$$\begin{bmatrix} r \\ j \end{bmatrix} = \begin{cases} 1 & \text{if } j = 0 \\ \dfrac{r(r-1)\cdots(r-j+1)}{1.2\cdots j} & \text{if } j > 0 \end{cases}$$

## APPENDIX B

# PROGRAMS

The following list includes the main programs and functions used in the simulation of controllers which are designed in the related chapters.

| | |
|---|---|
| LINSF_1.m: | State-feedback control simulation for the model linearized at start-up. |
| LIN.m | Classical output control (COC) simulation the model linearized at start-up. |
| CONT_1.m | COC simulation for the nonlinear model. |
| CONT_2.m | COC simulation with Kalman filter for the nonlinear model. |
| CSF_1.m | State feedback control simulation for the nonlinear model. |
| CSFSE_1.m | State feedback control with state estimation (SFCSE) via pole placement (Ackermann's formula) methodusing the nonlinear model in simulation. |
| CSFOPT_1.m | SFCSE simulation for the optimal state-feedback K (LQR) with deterministic linear observer using the nonlinear model. |
| LQG_1.m | Linear quadratic gaussian (LQG) simulation for the nonlinear model(Deterministic optimal state-feedback gain design+Kalman filter design). |
| LQG_2.m | LQG simulation using the nonlinear model and parameters which are dependent on opearation point $n_{ro}$(power level). |
| FC_1.m | Fuzzy logic controller(FLC) simulation for the nonlinear model. |
| FC_2.m | FLC simulation with different membership functions defined for error and change in error. |
| NC_1.m | Neurocontrol simulation for the nonlinear model. |
| ARE_SCH.m | function [X,Xresidua]=are_sch(K,R,Q) ARE solver using Schur decomposition. |
| ARE_2.m | function [X,Xresidua]=are2(L,K,R,Q) |

ARE solver usimg Newton iteration method.

FNNOUT.m function [y]=fnnout(E,DE,M,W)

Output of feedforward neural network defined in the matrices, M and W.

FUZZYOUT.m function[y]=fuzzyout(E,DE,P)

Output of FLC whose fuzzy rules are defined in the vector P.

FUZZYR_1.m function [P,X,Y,Fout,E,DE]=fuzzyr_1(Yc,Yd,Yp,X)

Fuzzy rule identification using least square estimation (LSE).

FUZZYR_2.m function [P,X,Y,Fout,E,DE]=fuzzyr_1(Yc,Yd,Yp,X)

Fuzzy rule identification using least square estimation (LSE) for the different membership functions for error E and change in error DE.

INTEG6BD.m function [X]=integ6bd(A,X,Q,h)

Sixth order BDFs for integration in the simulation.

INTEGBDF.m function [X]=integ6bd(A,X,Q,h,r)

rth order BDFs for integration where $1 \leq r \leq 6$.

MSHIP_1.m function [y]=mship(x,i)

1st Fuzzy membership function (proposed by J.Dombi) used by FC_1 and FC_2.

MSHIP_2.m function [y]=mship(x,i)

2nd Fuzzy membership function (proposed by J.Dombi) used bt FC_2.

TANH.m function [y]=tanh(x)

Sigmoid function used in the neural networks.

MLNNBP.m Backpropagation algorithm for multi-layered neural networks.

MLNNKF.m Learning algorithm via extended Kalman filter for multi-layered

neural networks.

MLKFBDF.m Learning algorithm via extended Kalman filter modified with BDFs for multi-layered neural networks.

EXPBDF.m Simulation test program for the integration of exponential inputs using BDFs.

SINBDF.m  Simulation test program for the integration of sinosodial inputs using BDFs.

```
% LINSF_1.m
% State-feedback control simulation for the linearized model at start up


Be=0.0065 ;
V=0.0001 ;
Gr=0.01  ;
Poa=2500;
mf=26.3 ;
ohm=6.53 ;
af=-0.00005 ;

L=0.125 ;
f=0.98;
Te=290 ;
mc=70.5;
M=92.8;
ac=0.00001;

A= [-Be/V     Be/V    af/V      ac/(2*V)      1/V;
      L      -L      0         0          0 ;
    f*Poa/mf  0   -ohm/mf    ohm/(2*mf)     0 ;
  (1-f)*Poa/mc 0     ohm/mc  -(2*M+ohm)/(2*mc)  0 ;
      0       0       0         0          0 ];
B=[0; 0; 0; 0; Gr];
C=[1 0 0 0 0];

X=[0 0 0 0 0]';
to=0;
h=0.1;
numofpoints=500;
gc=0.5;
Pd=0.1/3.181;
Bc=gc*B;
 Zr=0;
 T=to;
K=[-0.9895  -0.6267   0.0074  -0.0031  -96.1626]
Ac=A-Bc*C-Bc*K;
 for r=1:6,
  t=to+r*h;

   X=integbdf(Ac,X,Bc*Pd,h,r);
   T=[T t];

 end

i=8;
t=to+7*h;

while(i<numofpoints)

   X=integ6bd(Ac,X,Bc*Pd,h);
   T=[T t];
 t=t+h;
 i=i+1;
end

%LIN.m
%COC simulation for the linearized model at start-up.

Be=0.0065 ;
V=0.0001 ;
Gr=0.01  ;
Poa=2500;
mf=26.3 ;
ohm=6.53 ;
% no temperature feedback
af=-0.00005 ;

L=0.125 ;
f=0.98;
Te=290 ;
mc=70.5;
M=92.8;
ac=0.00001;
Pd=-0.9;

B= '[-Be/V     Be V    af/V      ac/(2*V)      1/V:      L      -L      0        0       0 :     f*Poa/mf  0
-ohm/mf    ohm (2*mf)     0 ;   (1-f)*Poa/mc 0    ohm mc  -(2*M+ohm)/(2*mc)  0 ;    -0.005       0
0         0        0 ]'
Q='[0:0 :0:0:0.005*Pd ]':


X(:,1)=[0 0 0 0 0]':
to=2.5:
T=[to]:
h=0.3:
m=5;
numofpoints=500:

t=to+h:
X(:,2)=-inv(-eye(m)+h*eval(B))*(X(:,1)+h*eval(Q)):
T=[T t]:
t=to+2*h:
X(:,3)=-inv(-(3/2)*eye(m)+h*eval(B))*(2*X(:,2)-(1/2)*X(:,1)+h*eval(Q));
T=[T t]:

t=to+3*h:
X(:,4)=-inv(-(11 6)*eye(m)+h*eval(B))*(3*X(:,3)-(3/2)*X(:,2)+(1/3)*X(:,1)+h*eval(Q));
T=[T t];

t=to+4*h:
X(:,5)=-inv(-(25 12)*eye(m)+h*eval(B))*(4*X(:,4)-3*X(:,3)+(4/3)*X(:,2)-(1/4)*X(:,1)+h*eval(Q));
T=[T t];

t=to+5*h:
X(:,6)=-inv(-(137 60)*eye(m)+h*eval(B))*(5*X(:,5)-5*X(:,4)+(10/3)*X(:,3)-
(5/4)*X(:,2)+(1 5)*X(:,1)+h*eval(Q));
T=[T t]:

t=to+6*h:
```

```
X(:,7)=-inv(-(147/60)*eye(m)+h*eval(B))*(6*X(:,6)-(15/2)*X(:,5)+(20/3)*X(:,4)-
(15/4)*X(:,3)+(6/5)*X(:,2)-(1/6)*X(:,1)+h*eval(Q));
T=[T t];

i=8;
t=to+7*h;
while(i<numofpoints)
X(:,i)=-inv(-(147/60)*eye(m)+h*eval(B))*(6*X(:,i-1)-(15/2)*X(:,i-2)+(20/3)*X(:,i-3)-(15/4)*X(:,i-
4)+(6/5)*X(:,i-5)-(1/6)*X(:,i-6)+h*eval(Q));
T=[T t];
t=t+h;
i=i+1;
end


% CONT_1.m
% Classical Output Unitary Feedback Control
% Non-linear Plant Model Simulationwith BDF's
% System Simulation used in "State feedback Assisted classical
% Control :An Incremental Approach to control modernization of
% Existing and future Nuclear Reactors and power plants"


 Be=0.0065 ;
 V=0.0001 ;
 Gr=0.01 ;
 Poa=2500;
 mf=26.3 ;
 ohm=6.53 ;
 af=-0.00005;
% af=0;
 L=0.125 ;
 f=0.98;
 Te=290 ;
 mc=70.5;
 M=92.8;
 ac=0.00001;
% ac=0;
 A1  = '[-L 0 0;0 -ohm/mf ohm/(2*mf);0 ohm/mc -((ohm/2)+M)/mc]';
 BTe= '[0 ; ohm*Te/(2*mf);(M-(ohm/2))*Te/mc]';
 B1 = '[L ; f*Poa/mf ; (1-f)*Poa/mc]';

 Pd=1.1;
 Gc=0.1585;
 Gc=0.5;
 X1(:,1)=[1 678.6607 316.9396]';
 Xnr(1)=[1];
 Xrr(1)=[2.704e-8];
 to=0;
 h=0.1;
 numofpoints=500;
 Zr=0;
 T=to;
 for r=1:6,
  t=to+r*h;
  X1=integbdf(eval(A1),X1,eval(B1)*Xnr(r)+eval(BTe),h,r);
  Q2=(1/V)*(((af*(X1(2,r)-X1(2,1))+ac*(X1(3,r)-X1(3,1)))+Xrr(r))*Xnr(r))+(Be/V)*X1(1,r);
  Xnr=integbdf(-Be/V,Xnr,Q2,h,r);
 Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2)- Xnr(r) );
% rod speed limitation
% Q3=max(-0.0125*Gr,min(Q3,0.0125*Gr));
 Zr=[Zr Q3/Gr];
 Xrr=integbdf(0,Xrr,Q3,h,r);
 T=[T t];
 end

i=8;
t=to+7*h;

while(i<numofpoints)

 X1=integ6bd(eval(A1),X1,eval(B1)*Xnr(i-1)+eval(BTe),h);
 Q2=(1/V)*(((af*(X1(2,i-1)-X1(2,1))+ac*(X1(3,i-1)-X1(3,1)))+Xrr(i-1))*Xnr(i-1))+(Be/V)*X1(1,i-1);
 Xnr=integ6bd(-Be/V,Xnr,Q2,h);
 Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2) - Xnr(i-1) );
% rod speed limitation
% Q3=max(-0.0125*Gr,min(Q3,0.0125*Gr));
 Zr=[Zr Q3/Gr];
 Xrr=integ6bd(0,Xrr,Q3,h);
 T=[T t];

 t=t+h;
 i=i+1;
end


%CONT_2.m
%Classical Output Power Feedback Control in Noisy Environment
%Output Power is disturbed with zero mean, white and gaussion noise
%Kalman Filter is used to estimate output power signal from noisy environment

% System Simulation used in "State feedback Assisted classical
% Control :An Incremental Approach to control modernization of
% Existing and future Nuclear Reactors and power plants"


 Be=0.0065 ;
 V=0.0001 ;
 Gr=0.01  ;
 Poa=2500;
 mf=26.3 ;
 ohm=6.53 ;
 af=-0.00005;
% af=0;
 L=0.125 ;
 f=0.98;
 Te=290 ;
 mc=70.5;
 M=92.8;
 ac=0.00001;
% ac=0;
 A1  = '[-L 0 0:0 -ohm/mf ohm/(2*mf);0 ohm/mc -((ohm/2)+M)/mc]';
 BTe= '[0 ; ohm*Te/(2*mf);(M-(ohm/2))*Te/mc]';
 B1 = '[L ; f*Poa/mf ; (1-f)*Poa/mc]';

%noise specifications
```

```
%Covariance of noise
%Rn=E[V*V']
Rn=-1;
% E[V]=0;
%P:covariance of (Xnr-Xne)
%initial value for P
P=10000;
%Xne

 Pd=1.1;
 Gc=0.1585;
 X1(:,1)=[1 678.6607 316.9396]';
 Xnr(1)=[1];
 Xne(1)=[1];
 Xrr(1)=[2.704e-8];
 to=0;
 h=0.1;
 numofpoints=1000;
 Zr=0;
 T=to;
 for r=1:6,
  t=to+r*h;

  yout(r)=Xnr(r)+randn(1)/10;
  Kgain=P*inv(P+Rn);
  P=(1-Kgain)*P;
  Xne=Xne+Kgain*(yout(r)-Xne);
  Xest=[Xest Xne];
  X1=integbdf(eval(A1),X1,eval(B1)*Xnr(r)+eval(BTe),h,r);
  Q2=(1/V)*(((af*(X1(2,r)-X1(2,1))+ac*(X1(3,r)-X1(3,1)))+Xrr(r))*Xnr(r))+(Be/V)*X1(1,r);
  Xnr=integbdf(-Be/V,Xnr,Q2,h,r);
  Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2)- Xne);
% rod speed limitation
% Q3=max(-0.0125*Gr,min(Q3,0.0125*Gr));
  Zr=[Zr Q3/Gr];
  Xrr=integbdf(0,Xrr,Q3,h,r);
  T=[T t];
 end

i=8;
t=to+7*h;

while(i<numofpoints)

  yout(i-1)=Xnr(i-1)+randn(1)/10;
  Kgain=P*inv(P+Rn);
  P=(1-Kgain)*P;
  Xne=Xne+Kgain*(yout(i-1)-Xne);
  Xest=[Xest Xne];


  X1=integ6bd(eval(A1),X1,eval(B1)*Xnr(i-1)+eval(BTe),h);
  Q2=(1/V)*(((af*(X1(2,i-1)-X1(2,1))+ac*(X1(3,i-1)-X1(3,1)))+Xrr(i-1))*Xnr(i-1))+(Be/V)*X1(1,i-1);
  Xnr=integ6bd(-Be/V,Xnr,Q2,h);
  Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2) - Xne);
% rod speed limitation
% Q3=max(-0.0125*Gr,min(Q3,0.0125*Gr));
  Zr=[Zr Q3/Gr];
 Xrr=integ6bd(0,Xrr,Q3,h);
 T=[T t];

t=t+h;
i=i+1;
end

%CSF_1.m
%State-feedback control simulation for nonlinear model

 Be=0.0065 ;
 V=0.0001 ;
 Gr=0.01  ;
 Poa=2500;
 mf=26.3 ;
 ohm=6.53 ;
 af=-0.00005;
 L=0.125 ;
 f=0.98;
 Te=290 ;
 mc=70.5;
 M=92.8;
 ac=0.00001;

 A1  = '[-L 0 0;0 -ohm/mf ohm/(2*mf);0 ohm/mc -((ohm/2)+M)/mc]';
 BTe= '[0 ; ohm*Te/(2*mf);(M-(ohm/2))*Te/mc]';
 B1 = '[L ; f*Poa/mf ; (1-f)*Poa/mc]';

 Pd=1+0.2/3.181;
 Gc=0.5;
 K=[-0.9895  -0.6267   0.0074  -0.0031  -96.1626]
 X1(:,1)=[1 678.6607 316.9396]';
 Xnr(1)=[1];
 Xrr(1)=[2.704e-8];
 to=0;
 h=0.1;
 numofpoints=500;
 Zr=0;
 T=to;
 for r=1:6,
  t=to+r*h;
  X1=integbdf(eval(A1),X1,eval(B1)*Xnr(r)+eval(BTe),h,r);
  Q2=(1/V)*(((af*(X1(2,r)-X1(2,1))+ac*(X1(3,r)-X1(3,1)))+Xrr(r))*Xnr(r))+(Be/V)*X1(1,r);
  Xnr=integbdf(-Be/V,Xnr,Q2,h,r);
      Q3=Gr*Gc*(   1.0+(Pd-1)*((sign(t-2.5)+1)/2)-   Xnr(r)-(K(1)*(Xnr(r)-Xnr(1))+K(2:4)*(X1(:,r)-
X1(:,1))+K(5)*(Xrr(r)-Xrr(1))) );
  Zr=[Zr Q3/Gc*Gr];
  Xrr=integbdf(0,Xrr,Q3,h,r);
  T=[T t];
 end

i=8;
t=to+7*h;

while(i<numofpoints)

 X1=integ6bd(eval(A1),X1,eval(B1)*Xnr(i-1)+eval(BTe),h);
 Q2=(1/V)*(((af*(X1(2,i-1)-X1(2,1))+ac*(X1(3,i-1)-X1(3,1)))+Xrr(i-1))*Xnr(i-1))+(Be/V)*X1(1,i-1);
```

```
  Xnr=integ6bd(-Be/V,Xnr,Q2,h);
   Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2) - Xnr(i-1)-(K(1)*(Xnr(i-1)-Xnr(1))+K(2:4)*(X1(:,i-1)-
X1(:,1))+K(5)*(Xrr(i-1)-Xrr(1))) );
  Zr=[Zr Q3/Gr*Gc];
  Xrr=integ6bd(0,Xrr,Q3,h);
  T=[T t];

 t=t+h;
 i=i+1;
end

% CSFSE_1.m
% State-Feedback Control with State-Estimation via Pole Placement
% System Simulation used in "State feedback Assisted classical
% Control :An Incremental Approach to control modernization of
% Existing and future Nuclear Reactors and power plants"


 Be=0.0065 ;
 V=0.0001 ;
 Gr=0.01  ;
 Poa=2500;
 mf=26.3 ;
 ohm=6.53 ;
 af=-0.00005;
 L=0.125 ;
 f=0.98;
 Te=290 ;
 mc=70.5;
 M=92.8;
 ac=0.00001;

 A= [-Be/V     Be/V    af/V      ac/(2*V)      1/V;
          L      -L      0          0            0 ;
     f*Poa/mf   0    -ohm/mf    ohm/(2*mf)      0 ;
   (1-f)*Poa/mc 0     ohm/mc  -(2*M+ohm)/(2*mc)  0 ;
           0      0       0          0           0 ];
B=[0; 0; 0; 0; Gr];
C=[1 0 0 0 0];

 A1  = '[-L 0 0;0 -ohm/mf ohm/(2*mf);0 ohm/mc -((ohm/2)-M)/mc]';
 BTe= '[0 ; ohm*Te/(2*mf);(M-(ohm/2))*Te/mc]';
 B1 = '[L : f*Poa/mf : (1-f)*Poa/mc]';

 Pd=1-0.2/3.1183;
 Gc=0.5;
 K=[-0.9895  -0.6267   0.0074  -0.0031  -96.1626]
 H=[-0.2248;
   -0.4933;
   93.1370;
    0.7743;
    0.0014]

 X1=[1 678.6607 316.9396]';
 Xe=[0 0 0 0 0]';
 Bc=Gc*B;
 Ac=A-Bc*C;
```

```
Xnr=[1];
Xrr=[2.704e-8];
to=0;
h=0.1;
numofpoints=500;
Zr=0;
T=to;
for r=1:6,
 t=to+r*h;
 X1=integbdf(eval(A1),X1,eval(B1)*Xnr(r)+eval(BTe),h,r);
 Q2=(1/V)*(((af*(X1(2,r)-X1(2,1))+ac*(X1(3,r)-X1(3,1)))+Xrr(r))*Xnr(r))+(Be/V)*X1(1,r);
 Xnr=integbdf(-Be/V,Xnr,Q2,h,r);
 Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2)- Xnr(r)-K*(Xe(:,r)-Xe(:,1)));
 Zr=[Zr Q3/(Gc*Gr)];
 Xrr=integbdf(0,Xrr,Q3,h,r);
 Xe=integbdf(Ac-H*C,Xe,Bc*(Xnr(r)-1+Q3/(Gc*Gr))+H*(Xnr(r)-1),h,r);

 T=[T t];
end

i=8;
t=to+7*h;

while(i<numofpoints)

 X1=integ6bd(eval(A1),X1,eval(B1)*Xnr(i-1)+eval(BTe),h);
 Q2=(1/V)*(((af*(X1(2,i-1)-X1(2,1))+ac*(X1(3,i-1)-X1(3,1)))+Xrr(i-1))*Xnr(i-1))+(Be/V)*X1(1,i-1);
 Xnr=integ6bd(-Be/V,Xnr,Q2,h);
 Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2) - Xnr(i-1)-K*(Xe(:,i-1)-Xe(:,1)) );
 Zr=[Zr Q3/(Gr*Gc)];
 Xrr=integ6bd(0,Xrr,Q3,h);
 Xe=integ6bd(Ac-H*C,Xe,Bc*(Xnr(i-1)-1+Q3/(Gr*Gc))+H*(Xnr(i-1)-1),h);

 T=[T t];

 t=t+h;
 i=i+1;
end

% CSFOPT_1.m
% Optimal Controller Design(LQR)

% System Simulation used in "State feedback Assisted classical
% Control :An Incremental Approach to control modernization of
% Existing and future Nuclear Reactors and power plants"


 Be=0.0065 ;
 V=0.0001 ;
 Gr=0.01  ;
 Poa=2500;
 mf=26.3 ;
 ohm=6.53 ;
 af=-0.00005;
 L=0.125 ;
 f=0.98;
 Te=290 ;
 mc=70.5;
 M=92.8;
 ac=0.00001;

 A= [-Be/V     Be/V   af/V      ac/(2*V)      1/V;
          L       -L      0          0           0 ;
     f*Poa/mf  0   -ohm/mf    ohm/(2*mf)     0 ;
   (1-f)*Poa/mc 0     ohm/mc  -(2*M+ohm)/(2*mc)  0 ;
           0       0       0          0          0 ];
B=[0; 0; 0; 0; Gr];
C=[1 0 0 0 0];

 A1  = '[-L 0 0;0 -ohm/mf ohm/(2*mf);0 ohm/mc -((ohm/2)+M)/mc]';
 BTe= '[0 ; ohm*Te/(2*mf);(M-(ohm/2))*Te/mc]';
 B1 = '[L ; f*Poa/mf ; (1-f)*Poa/mc]';
%for state feedback pole placement method v=1/3.1183
%Pd=1+0.1/3.1183:
 Gc=0.5:
% State feedback gain found using pole placement method
%(deterministic case Ackermann's formula)
% K=[-0.9895  -0.6267   0.0074  -0.0031  -96.1626]
% calculating optimal gain
 r=3000;
 Q=[0 0 0 0 0;0 0 0 0 0;0 0 0.1 0 0;0 0 0 0.01 0;0 0 0 0 0];
% Solving ARE using Newton Iteration for Stiff Case
 [P,res]=are2((A-Gc*B*C)',A-Gc*B*C,Gc*B*inv(r)*B'*Gc',Q);
% Solving ARE via Schur Decomposition Method
% [P,res]=are_sch(A-Gc*B*C,Gc*B*inv(r)*B'*Gc',Q)
 K=inv(r)*B'*Gc'*P
% Deterministic observer gain observer gain (using Ackermann's formula)
 H=[-0.2248;
   -0.4933;
   93.1370;
    0.7743;
    0.0014]
%calculating nondynamical compensator gain to track input for unit step response
[z,p,k]=ss2zp(A-Gc*B*(C+K),Gc*B,C,0,1);
[np]=size(p,1);
[nz]=size(z,1);
pm=1;
for i=1:nz, pm=pm*z(i);end
for i=1:np, pm=pm.p(i);end
v=1/(k*pm)
Pd=1+0.1*v;
Pdd1=1;
Pdd=1.1;
 X1=[1 678.6607 316.9396]';
 Xe=[0 0 0 0 0]';
 Bc=Gc*B;
 Ac=A-Bc*C;

 Xnr=[1];
 Xrr=[2.704e-8];
 to=0;
 h=0.1;
 numofpoints=500;
 Zr=0;
 T=to;
 for r=1:6,
  t=to+r*h;
  X1=integbdf(eval(A1),X1,eval(B1)*Xnr(r)+eval(BTe),h,r);
  Q2=(1/V)*(((af*(X1(2,r)-X1(2,1))+ac*(X1(3,r)-X1(3,1)))+Xrr(r))*Xnr(r))+(Be/V)*X1(1,r);
  Xnr=integbdf(-Be/V,Xnr,Q2,h,r);
  Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2)- Xnr(r)-K*(Xe(:,r)-Xe(:,1)));
% rod speed limitation
%  Q3=max(-0.0125*Gr,min(Q3,0.0125*Gr));
 Pdd1=[Pdd1 1.0+(Pdd-1)*((sign(t-2.5)+1)/2)];
  Zr=[Zr Q3/Gr];
  Xrr=integbdf(0,Xrr,Q3,h,r);
  Xe=integbdf(Ac-H*C,Xe,Bc*(Xnr(r)-1+Q3/(Gc*Gr))+H*(Xnr(r)-1),h,r);

  T=[T t];
 end

i=8;
t=to+7*h;

while(i<numofpoints)

  X1=integ6bd(eval(A1),X1,eval(B1)*Xnr(i-1)+eval(BTe),h);
  Q2=(1/V)*(((af*(X1(2,i-1)-X1(2,1))+ac*(X1(3,i-1)-X1(3,1)))+Xrr(i-1))*Xnr(i-1))+(Be/V)*X1(1,i-1);
  Xnr=integ6bd(-Be/V,Xnr,Q2,h);
  Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2) - Xnr(i-1)-K*(Xe(:,i-1)-Xe(:,1)) );
 Pdd1=[Pdd1 1.0+(Pdd-1)*((sign(t-2.5)+1)/2)];

% rod speed limitation
%  Q3=max(-0.0125*Gr,min(Q3,0.0125*Gr));

  Zr=[Zr Q3/Gr];
  Xrr=integ6bd(0,Xrr,Q3,h);
  Xe=integ6bd(Ac-H*C,Xe,Bc*(Xnr(i-1)-1+Q3/(Gr*Gc))+H*(Xnr(i-1)-1),h);

  T=[T t];

 t=t+h;
 i=i+1;
end

%LQG_2
% Linear Quadratic Gaussion (LQG)
% Robust Conrol Design Methodology


 Be=0.0065 ;
 V=0.0001 ;
 Gr=0.01  ;
 Poa=2500;
 mf=26.3 ;
 ohm=6.53 ;
 af=-0.00005;
 L=0.125 ;
 f=0.98;
 Te=290 ;
 mc=70.5;
 M=92.8;
 ac=0.00001;
 A= [-Be/V     Be/V   af/V     ac/(2*V)      1/V;
          L      -L      0          0          0 ;
    f*Poa/mf  0   -ohm/mf    ohm/(2*mf)      0 ;
   (1-f)*Poa/mc 0     ohm/mc  -(2*M+ohm)/(2*mc)  0 ;
          0      0      0          0          0 ];
B=[0; 0: 0: 0; Gr];
C=[1 0 0 0 0];

 A1  = '[-L 0 0;0 -ohm/mf ohm/(2*mf);0 ohm/mc -((ohm/2)+M)/mc]';
 BTe= '[0 : ohm*Te/(2*mf);(M-(ohm/2))*Te/mc]';
 B1 = '[L : f*Poa/mf ; (1-f)*Poa/mc]';
%for state feedback pole placement method v=1/3.1183
%Pd=1+0.1/3.1183;
 Gc=0.5:
% State feedback gain found using pole placement method
%(deterministic case Ackermann's formula)
% K=[-0.9895   -0.6267   0.0074   -0.0031  -96.1626]
% calculating optimal gain
 r=3000:
 Q=[0 0 0 0 0:0 0 0 0 0;0 0 0.1 0 0:0 0 0 0.01 0:0 0 0 0 0];
 [P,res]=are2((A-Gc*B*C)'.A-Gc*B*C.Gc*B*inv(r)*B'*Gc',Q);
 K=inv(r)*B'*Gc'*P
% Deterministic observer gain observer gain (using Ackermann's formula)
% H=[-0.2248:
%    -0.4933:
%    93.1370:
%     0.7743:
%     0.0014]

 re=50000:
 Qe=[K(1)^2 0 0 0 0;0 K(2)^2 0 0 0;0 0 K(3)^2 0 0:0 0 0 K(4)^2 0:0 0 0 0 K(5)^2]:
 [Pe,Rese]=are2((A-Gc*B*C),(A-Gc*B*C)',C'*inv(re)*C,Qe);
 H=-Pe*C'*inv(r)

%
%calculating nondynamical compensator gain to track input for unit step response
[z,p,k]=ss2zp(A-Gc*B*(C+K),Gc*B,C,0,1);
[np]=size(p,1);
[nz]=size(z,1);
pm=1:
for i=1:nz, pm=pm*z(i);end
for i=1:np, pm=pm/p(i);end
v=1/(k*pm)
Pd=1+0.1*v;
Pdd1=1:
Pdd=1.1:
 X1=[1 678.6607 316.9396]':
 Xe=[0 0 0 0 0]';
 Bc=Gc*B;
 Ac=A-Bc*C:

 Xnr=[1]:
 Xrr=[2.704e-8]:
 to=0;
 h=0.1:
 numofpoints=250:
 Zr=0;
 T=to;
 for r=1:6,
  t=to+r*h;
  X1=integbdf(eval(A1),X1,eval(B1)*Xnr(r)+eval(BTe),h,r);
  Q2=(1/V)*(((af*(X1(2,r)-X1(2,1))+ac*(X1(3,r)-X1(3,1)))+Xrr(r))*Xnr(r))+(Be/V)*X1(1,r);
  Xnr=integbdf(-Be/V,Xnr,Q2,h,r);
  Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2)- Xnr(r)-K*(Xe(:,r)-Xe(:,1)));
% rod speed limitation
%  Q3=max(-0.0125*Gr,min(Q3,0.0125*Gr));
 Pdd1=[Pdd1 1.0+(Pdd-1)*((sign(t-2.5)+1)/2)];
  Zr=[Zr Q3/Gr];
  Xrr=integbdf(0,Xrr,Q3,h,r);
  Xe=integbdf(Ac-H*C,Xe,Bc*(Xnr(r)-1+Q3/(Gc*Gr))+H*(Xnr(r)-1),h,r);

  T=[T t];
 end

i=8;
t=to+7*h;

while(i<numofpoints)

  X1=integ6bd(eval(A1),X1,eval(B1)*Xnr(i-1)+eval(BTe),h);
  Q2=(1/V)*(((af*(X1(2,i-1)-X1(2,1))+ac*(X1(3,i-1)-X1(3,1)))+Xrr(i-1))*Xnr(i-1))+(Be/V)*X1(1,i-1);
  Xnr=integ6bd(-Be/V,Xnr,Q2,h);
  Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2) - Xnr(i-1)-K*(Xe(:,i-1)-Xe(:,1)) );
 Pdd1=[Pdd1 1.0+(Pdd-1)*((sign(t-2.5)+1)/2)];

% rod speed limitation
%  Q3=max(-0.0125*Gr,min(Q3,0.0125*Gr));

  Zr=[Zr Q3/Gr];
  Xrr=integ6bd(0,Xrr,Q3,h);
  Xe=integ6bd(Ac-H*C,Xe,Bc*(Xnr(i-1)-1+Q3/(Gr*Gc))+H*(Xnr(i-1)-1),h);

  T=[T t];

 t=t+h;
 i=i+1;
end


%LQG_2
% Linear Quadratic Gaussion (LQG)
% Robust Conrol Design Methodology

 Be=0.0065 ;
 V=0.0001 ;
 Gr=0.01  ;
 Poa=2500;
 mf=26.3 ;
 ohm=6.53 ;
 af=-0.00005;
 L=0.125 ;
 f=0.98;
 Te=290 ;
 mc=70.5;
 M=92.8;
 ac=0.00001;
 A= [-Be/V     Be/V    af/V      ac/(2*V)      1/V;
          L      -L      0         0           0 ;
    f*Poa/mf  0   -ohm/mf    ohm/(2*mf)     0 ;
   (1-f)*Poa/mc 0     ohm/mc  -(2*M+ohm)/(2*mc)  0 ;
           0      0       0         0          0 ];
B=[0; 0; 0; 0; Gr];
C=[1 0 0 0 0];

 A1  = '[-L 0 0;0 -ohm/mf ohm/(2*mf);0 ohm/mc -((ohm/2)+M)/mc]';
 BTe= '[0 ; ohm*Te/(2*mf);(M-(ohm/2))*Te/mc]';
 B1 = '[L ; f*Poa/mf ; (1-f)*Poa/mc]';
%for state feedback pole placement method v=1/3.1183
%Pd=1+0.1/3.1183;
 Gc=0.5;
% State feedback gain found using pole placement method
%(deterministic case Ackermann's formula)
% K=[-0.9895  -0.6267   0.0074  -0.0031  -96.1626]
% calculating optimal gain
 r=3000;
 Q=[0 0 0 0 0;0 0 0 0 0;0 0 0.1 0 0;0 0 0 0.01 0;0 0 0 0 0];
 [P,res]=are2((A-Gc*B*C)',A-Gc*B*C,Gc*B*inv(r)*B'*Gc',Q);
 K=inv(r)*B'*Gc'*P
% Deterministic observer gain  (using Ackermann's formula)
% H=[-0.2248;
%    -0.4933;
%    93.1370;
%     0.7743;
%     0.0014]

 re=50000;
 Qe=[K(1)^2 0 0 0 0;0 K(2)^2 0 0 0;0 0 K(3)^2 0 0;0 0 0 K(4)^2 0;0 0 0 0 K(5)^2];
 [Pe,Rese]=are2((A-Gc*B*C),(A-Gc*B*C)',C'*inv(re)*C,Qe);
 H=-Pe*C'*inv(r)

%calculating nondynamical compensator gain to track input for unit step response
[z,p,k]=ss2zp(A-Gc*B*(C+K),Gc*B,C,0,1);
[np]=size(p,1);
[nz]=size(z,1);
pm=1;
for i=1:nz, pm=pm*z(i);end
for i=1:np, pm=pm/p(i);end
v=1/(k*pm)
Pd=1+0.1*v;
Pdd1=1;
Pdd=1.1;
 X1=[1 678.6607 316.9396]';
 Xe=[0 0 0 0 0]';
 Bc=Gc*B;
 Ac=A-Bc*C;

 Xnr=[1];
 Xrr=[2.704e-8];
 to=0;
 h=0.1;
 numofpoints=250;
 Zr=0;
 T=to;
 for r=1:6,
  t=to+r*h;
  X1=integbdf(eval(A1),X1,eval(B1)*Xnr(r)+eval(BTe),h,r);
  Q2=(1/V)*(((af*(X1(2,r)-X1(2,1))+ac*(X1(3,r)-X1(3,1)))+Xrr(r))*Xnr(r))+(Be/V)*X1(1,r);
  Xnr=integbdf(-Be/V,Xnr,Q2,h,r);
  Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2)- Xnr(r)-K*(Xe(:,r)-Xe(:,1)));
% rod speed limitation
%  Q3=max(-0.0125*Gr,min(Q3,0.0125*Gr));
 Pdd1=[Pdd1 1.0+(Pdd-1)*((sign(t-2.5)+1)/2)];
  Zr=[Zr Q3/Gr];
  Xrr=integbdf(0,Xrr,Q3,h,r);
  Xe=integbdf(Ac-H*C,Xe,Bc*(Xnr(r)-1+Q3/(Gc*Gr))+H*(Xnr(r)-1),h,r);

  T=[T t];
 end

i=8;
t=to+7*h;

while(i<numofpoints)

  X1=integ6bd(eval(A1),X1,eval(B1)*Xnr(i-1)+eval(BTe),h);
  Q2=(1/V)*(((af*(X1(2,i-1)-X1(2,1))+ac*(X1(3,i-1)-X1(3,1)))+Xrr(i-1))*Xnr(i-1))+(Be/V)*X1(1,i-1);
  Xnr=integ6bd(-Be/V,Xnr,Q2,h);
  Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2) - Xnr(i-1)-K*(Xe(:,i-1)-Xe(:,1)) );
 Pdd1=[Pdd1 1.0+(Pdd-1)*((sign(t-2.5)+1)/2)];

% rod speed limitation
%  Q3=max(-0.0125*Gr,min(Q3,0.0125*Gr));

  Zr=[Zr Q3/Gr];
  Xrr=integ6bd(0,Xrr,Q3,h);
  Xe=integ6bd(Ac-H*C,Xe,Bc*(Xnr(i-1)-1+Q3/(Gr*Gc))+H*(Xnr(i-1)-1),h);

  T=[T t];

 t=t+h;
 i=i+1;
end

%LQG_2
% Linear Quadratic Gaussion (LQG)
% Robust Conrol Design Methodology
% Operating point dependent parameters are included in non-linear model
% Optimal Stochastic Observer Design with Optimal State Feedback Gain
% (time invariant case Kalman Filter Design)

% System Simulation Parameters are taken from
% R.M.Edwards et. al., Robust Optimal Control of Nuclear Reactors and Power Plants.
%                  Nuclear Technology Vol.98 May 1992,p.137-148.
% A.Ben-Abdennour. et. al., LQG/LTR Robust Control ofNuclera Reactors with
%                  Improved Temperature Performance. IEEE Trans.on
%                 Nuclear Science Vol.39,No.6 December 1992


%   Reactors with Improved Temperature Performance"
% IEEE Tran. on Nuclear Science Vol.39,No.6,December 1992 pg 2286-2294
% Parameters for Controller Design at the Middle of Fuel Cycle of
% a TMI-Type PWR

 Be=0.006019     ; % fraction of delayed fission neutrons
 V=0.00002       ; % effective prompt neutron life time (s)
 Gr=0.01450      ; % reactivity worth of rod per unit length
 Poa=2500        ; % initial equlibrium power lever (MW)
 mf=26.3         ; % total heat capacity of fuel and structural material (MW.s/C)
 ohm=6.6         ; % heat transfer coeffient between fuel and coolant
 af=-0.0000324   ; % fuel temperature reactivity coeffient
 L=0.150         ; % effective precursor radioactive decay constant (1/s)
 f=0.92          ; % fraction of reactor power deposited in fuel
 Te=290          ; % temperature of the water entering the reactor (C)
 mc=71.8         ; % total heat capacity of reactor coolant (MW.s/C)
 M =102.0        ; % mass flow rate times heat capacity of the water (MW/C)
 ac=-0.000213    ; % coolant temperature reactivity coefficient

 A= [-Be/V     Be/V    af/V      ac/(2*V)      1/V;
          L      -L      0         0           0 ;
    f*Poa/mf  0   -ohm/mf    ohm/(2*mf)      0 ;
  (1-f)*Poa/mc 0    ohm/mc  -(2*M+ohm)/(2*mc)  0 ;
          0      0      0         0           0 ];
B=[0; 0; 0; 0; Gr];
C=[1 0 0 0 0];

 A1  = '[-L 0 0;0 -ohm0/mf ohm0/(2*mf);0 ohm0/mc0 -((ohm0/2)+M0)/mc0]';
 BTe= '[0 ; ohm0*Te/(2*mf);(M0-(ohm0/2))*Te/mc0]';
 B1 = '[L ; f*Poa/mf ; (1-f)*Poa/mc0]';
Gc=0.5;
% calculating optimal gain
 r=3000;
 Q=[0 0 0 0 0;0 0 0 0 0;0 0 0.1 0 0;0 0 0 0.01 0;0 0 0 0 0];
 [P,res]=are2((A-Gc*B*C)',A-Gc*B*C,Gc*B*inv(r)*B'*Gc',Q);
 K=inv(r)*B'*Gc'*P
%designing Kalman filter
re=30000;
 Q=[K(1)^2 0 0 0 0;0 K(2)^2 0 0 0;0 0 K(3)^2 0 0;0 0 0 K(4)^2 0;0 0 0 0 K(5)^2];
 [Pe,Rese]=are2((A-Gc*B*C),(A-Gc*B*C)',C'*inv(re)*C,Q);
 H=-Pe*C'*inv(r)

%
%calculating nondynamical compensator gain to track input for unit step response
[z,p,k]=ss2zp(A-Gc*B*(C+K),Gc*B,C,0,1);
[np]=size(p,1);
[nz]=size(z,1);
pm=1;
for i=1:nz, pm=pm*z(i);end
for i=1:np, pm=pm/p(i);end
v=1/(k*pm)
Pd=1-0.5*v;
Pdd1=1;
Pdd=0.5;
% X1=[1 678.6607 316.9396]';
  X1=[1 650.7415 314.5098]';
Xe=[0 0 0 0 0]';
 Bc=Gc*B;
 Ac=A-Bc*C;
 Xnr=[1];
 Xrr=[1.174828e-9];
 to=0;
 h=0.1;
 numofpoints=500;
 Zr=0;
 T=to;
 for r=1:6,
 nro=Xnr(r);
% parameters depend on operating point nro;
 af=(nro-4.24)*1e-5;
 ac=(-4*nro-17.3)*1e-5;
 mc0=((160/9)*nro+54.022);
 ohm0=((5/3)*nro+4.9333);
 M0=(28*nro+74);

  t=to+r*h;
  X1=integbdf(eval(A1),X1,eval(B1)*Xnr(r)+eval(BTe),h,r);
  Q2=(1/V)*(((af*(X1(2,r)-X1(2,1))+ac*(X1(3,r)-X1(3,1)))+Xrr(r))*Xnr(r))+(Be/V)*X1(1,r);
  Xnr=integbdf(-Be/V,Xnr,Q2,h,r);
  Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2)- Xnr(r)-K*(Xe(:,r)-Xe(:,1)));
% rod speed limitation
%  Q3=max(-0.0125*Gr,min(Q3,0.0125*Gr));
 Pdd1=[Pdd1 1.0+(Pdd-1)*((sign(t-2.5)+1)/2)];
  Zr=[Zr Q3/Gr];
  Xrr=integbdf(0,Xrr,Q3,h,r);
  Xe=integbdf(Ac-H*C,Xe,Bc*(Xnr(r)-1+Q3/(Gc*Gr))+H*(Xnr(r)-1),h,r);

  T=[T t];
 end

i=8;
t=to+7*h;

while(i<numofpoints)


 nro=Xnr(i-1);
% parameters depend on operating point nro;
 af=(nro-4.24)*1e-5;
 ac=(-4*nro-17.3)*1e-5;
 mc0=((160/9)*nro+54.022);
 ohm0=((5/3)*nro+4.9333);
 M0=(28*nro+74);


  X1=integ6bd(eval(A1),X1,eval(B1)*Xnr(i-1)+eval(BTe),h);
  Q2=(1/V)*(((af*(X1(2,i-1)-X1(2,1))+ac*(X1(3,i-1)-X1(3,1)))+Xrr(i-1))*Xnr(i-1))+(Be/V)*X1(1,i-1);
  Xnr=integ6bd(-Be/V,Xnr,Q2,h);
  Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2) - Xnr(i-1)-K*(Xe(:,i-1)-Xe(:,1)) );
 Pdd1=[Pdd1 1.0+(Pdd-1)*((sign(t-2.5)+1)/2)];

% rod speed limitation
%  Q3=max(-0.0125*Gr,min(Q3,0.0125*Gr));

  Zr=[Zr Q3/Gr];
  Xrr=integ6bd(0,Xrr,Q3,h);
  Xe=integ6bd(Ac-H*C,Xe,Bc*(Xnr(i-1)-1+Q3/(Gr*Gc))+H*(Xnr(i-1)-1),h);
 T=[T t];

 t=t+h;
 i=i+1;
end

% FC_1.m
% Fuzzy Logic Controller Design

% System Simulation used in "State feedback Assisted classical
% Control :An Incremental Approach to control modernization of
% Existing and future Nuclear Reactors and power plants"


 Be=0.0065 ;
 V=0.0001 ;
 Gr=0.01  ;
 Poa=2500;
 mf=26.3 ;
 ohm=6.53 ;
 af=-0.00005;
 L=0.125 ;
 f=0.98;
 Te=290 ;
 mc=70.5;
 M=92.8;
 ac=0.00001;

 A= [-Be/V     Be/V    af/V      ac/(2*V)      1/V;
          L      -L       0         0           0 ;
    f*Poa/mf   0    -ohm/mf    ohm/(2*mf)      0 ;
  (1-f)*Poa/mc 0     ohm/mc  -(2*M+ohm)/(2*mc)  0 ;
          0       0       0         0           0 ];
B=[0; 0; 0; 0; Gr];
C=[1 0 0 0 0];

 A1  = '[-L 0 0;0 -ohm/mf ohm/(2*mf);0 ohm/mc -((ohm/2)+M)/mc]';
 BTe= '[0 ; ohm*Te/(2*mf);(M-(ohm/2))*Te/mc]';
 B1 = '[L ; f*Poa/mf ; (1-f)*Poa/mc]';
Gc=0.5;
Pd=1.1;
 X1=[1 678.6607 316.9396]';
 Xe=[0 0 0 0 0]';
 E=0;
 DE=0;
 Yfc=0;
 Bc=Gc*B;
 Ac=A-Bc*C;

 Xnr=[1];
 Xrr=[2.704e-8];
 to=0;
 h=0.01;
 numofpoints=1000;
 Zr=0;
 T=to;
 for r=1:6,
  t=to+r*h;
  X1=integbdf(eval(A1),X1,eval(B1)*Xnr(r)+eval(BTe),h,r);
  Q2=(1/V)*(((af*(X1(2,r)-X1(2,1))+ac*(X1(3,r)-X1(3,1)))+Xrr(r))*Xnr(r))+(Be/V)*X1(1,r);
  Xnr=integbdf(-Be/V,Xnr,Q2,h,r);
% calculation of error and change in error
  E=[E (Xnr(r)-(1.0+(Pd-1)*((sign(t-2.5)+1)/2)))]
  DE=[DE E(r+1)-E(r)];
  Bfc=1;
  for i=1:5,
   for j=1:5,
    Bfc=[Bfc mship_1(E(r),i)*mship_1(DE(r),j)];
   end
  end
  Bfc=Bfc(2:size(Bfc,2)) /(Bfc(2:size(Bfc,2))*ones(size(Bfc,2)-1,1));
  Xfc=[Bfc*E(r) Bfc*DE(r)];
% Fuzzy controller output
  Yfc=[Yfc Xfc*Pfr];

  Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2)- Xnr(r)+Yfc(r+1))

% rod speed limitation
%  Q3=max(-0.0125*Gr,min(Q3,0.0125*Gr));

  Zr=[Zr Q3/Gr];
  Xrr=integbdf(0,Xrr,Q3,h,r);
  Xe=integbdf(Ac-H*C,Xe,Bc*(Xnr(r)-1+Q3/(Gc*Gr))+H*(Xnr(r)-1),h,r);

  T=[T t];
 end

i=8;
t=to+7*h;

while(i<numofpoints)

  X1=integ6bd(eval(A1),X1,eval(B1)*Xnr(i-1)+eval(BTe),h);
  Q2=(1/V)*(((af*(X1(2,i-1)-X1(2,1))+ac*(X1(3,i-1)-X1(3,1)))+Xrr(i-1))*Xnr(i-1))+(Be/V)*X1(1,i-1);
  Xnr=integ6bd(-Be/V,Xnr,Q2,h);
% calculation of error and change in error
  E=[E (Xnr(i-1)-(1.0+(Pd-1)*((sign(t-2.5)+1)/2)))];
  DE=[DE E(i)-E(i-1)];
  Bfc=1;
  for ii=1:5,
   for j=1:5,
    Bfc=[Bfc mship_1(E(i-1),ii)*mship_1(DE(i-1),j)];
   end
  end
  Bfc=Bfc(2:size(Bfc,2)) /(Bfc(2:size(Bfc,2))*ones(size(Bfc,2)-1,1));
  Xfc=[Bfc*E(i-1) Bfc*DE(i-1)];
% Fuzzy controller output
  Yfc=[Yfc Xfc*Pfr];
  i
  Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2)- Xnr(i-1)+Yfc(i));
Xnr(i-1)

% rod speed limitation
%  Q3=max(-0.0125*Gr,min(Q3,0.0125*Gr));
```

```
  Zr=[Zr Q3/Gr];
  Xrr=integ6bd(0,Xrr,Q3,h);
  Xe=integ6bd(Ac-H*C,Xe,Bc*(Xnr(i-1)-1+Q3/(Gr*Gc))+H*(Xnr(i-1)-1),h);

  T=[T t];

 t=t+h;
 i=i+1;
end

% FC_2.m
% Fuzzy Logic Controller Design

% System Simulation used in "State feedback Assisted classical
% Control :An Incremental Approach to control modernization of
% Existing and future Nuclear Reactors and power plants"


 Be=0.0065 ;
 V=0.0001 ;
 Gr=0.01  ;
 Poa=2500;
 mf=26.3 ;
 ohm=6.53 ;
 af=-0.00005;
 L=0.125 ;
 f=0.98;
 Te=290 ;
 mc=70.5;
 M=92.8;
 ac=0.00001;

 A= [-Be/V     Be/V    af/V       ac/(2*V)       1/V;
          L       -L       0           0            0 ;
    f*Poa/mf  0    -ohm/mf     ohm/(2*mf)      0 ;
  (1-f)*Poa/mc 0     ohm/mc  -(2*M+ohm)/(2*mc)  0 ;
           0       0        0           0           0 ];
B=[0; 0; 0; 0; Gr];
C=[1 0 0 0 0];

 A1  = '[-L 0 0;0 -ohm mf ohm/(2*mf);0 ohm/mc -((ohm/2)+M)/mc]';
 BTe= '[0 ; ohm*Te/(2*mf);(M-(ohm/2))*Te/mc]';
 B1 = '[L ; f*Poa/mf ; (1-f)*Poa/mc]';
Gc=0.5;
Pd=1.1;
 X1=[1 678.6607 316.9396]';
 Xe=[0 0 0 0 0]';
 E=0;
 DE=0;
 Yfc=0;
 Bc=Gc*B;
 Ac=A-Bc*C;

 Xnr=[1];
 Xrr=[2.704e-8];
 to=0;
 h=0.01;
 numofpoints=1000;
 Zr=0;
 T=to;
 for r=1:6,
  t=to+r*h;
  X1=integbdf(eval(A1),X1,eval(B1)*Xnr(r)+eval(BTe),h,r);
  Q2=(1/V)*(((af*(X1(2,r)-X1(2,1))+ac*(X1(3,r)-X1(3,1)))+Xrr(r))*Xnr(r))+(Be/V)*X1(1,r);
  Xnr=integbdf(-Be/V,Xnr,Q2,h,r);
% calculation of error and change in error
  E=[E (Xnr(r)-(1.0+(Pd-1)*((sign(t-2.5)+1)/2)))]
  DE=[DE E(r+1)-E(r)];
  Bfc=1;
  for i=1:5,
    for j=1:5,
     Bfc=[Bfc mship_1(E(r),i)*mship_2(DE(r),j)];
    end
   end
   Bfc=Bfc(2:size(Bfc,2)) /(Bfc(2:size(Bfc,2))*ones(size(Bfc,2)-1,1));
   Xfc=[Bfc*E(r) Bfc*DE(r)];
% Fuzzy controller output
   Yfc=[Yfc Xfc*Pfr];

 Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2)- Xnr(r)+Yfc(r+1))

% rod speed limitation
%  Q3=max(-0.0125*Gr,min(Q3,0.0125*Gr));

 Zr=[Zr Q3/Gr];
 Xrr=integbdf(0,Xrr,Q3,h,r);
 Xe=integbdf(Ac-H*C,Xe,Bc*(Xnr(r)-1+Q3/(Gc*Gr))+H*(Xnr(r)-1),h,r);

 T=[T t];
end

i=8;
t=to+7*h;

while(i<numofpoints)

 X1=integ6bd(eval(A1),X1,eval(B1)*Xnr(i-1)+eval(BTe),h);
 Q2=(1/V)*(((af*(X1(2,i-1)-X1(2,1))+ac*(X1(3,i-1)-X1(3,1)))+Xrr(i-1))*Xnr(i-1))+(Be/V)*X1(1,i-1);
 Xnr=integ6bd(-Be/V,Xnr,Q2,h);
% calculation of error and change in error
  E=[E (Xnr(i-1)-(1.0+(Pd-1)*((sign(t-2.5)+1)/2)))];
 DE=[DE E(i)-E(i-1)];
 Bfc=1;
 for ii=1:5,
   for j=1:5,
    Bfc=[Bfc mship_1(E(i-1),ii)*mship_2(DE(i-1),j)];
   end
  end
  Bfc=Bfc(2:size(Bfc,2)) /(Bfc(2:size(Bfc,2))*ones(size(Bfc,2)-1,1));
  Xfc=[Bfc*E(i-1) Bfc*DE(i-1)];
% Fuzzy controller output
  Yfc=[Yfc Xfc*Pfr];
 i
 Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2)- Xnr(i-1)+Yfc(i));
Xnr(i-1)
```

```
% rod speed limitation
%  Q3=max(-0.0125*Gr,min(Q3,0.0125*Gr));

 Zr=[Zr Q3/Gr];
 Xrr=integ6bd(0,Xrr,Q3,h);
 Xe=integ6bd(Ac-H*C,Xe,Bc*(Xnr(i-1)-1+Q3/(Gr*Gc))+H*(Xnr(i-1)-1),h);

 T=[T t];

 t=t+h;
 i=i+1;
end

% NC_1.m
% Neurocontrol of Nuclear Power Plant

% System Simulation used in "State feedback Assisted classical
% Control :An Incremental Approach to control modernization of
% Existing and future Nuclear Reactors and power plants"


 Be=0.0065 ;
 V=0.0001 ;
 Gr=0.01  ;
 Poa=2500;
 mf=26.3 ;
 ohm=6.53 ;
 af=-0.00005;
 L=0.125 ;
 f=0.98;
 Te=290 ;
 mc=70.5;
 M=92.8;
 ac=0.00001;

 A= [-Be/V     Be/V    af/V      ac/(2*V)      1/V;
          L      -L      0         0          0 ;
    f*Poa/mf  0    -ohm/mf    ohm/(2*mf)      0 ;
  (1-f)*Poa/mc 0    ohm/mc  -(2*M+ohm)/(2*mc)  0 ;
          0      0       0         0          0 ];
B=[0; 0; 0; 0; Gr];
C=[1 0 0 0 0];

 A1  = '[-L 0 0;0 -ohm/mf ohm/(2*mf);0 ohm/mc -((ohm/2)+M)/mc]';
 BTe= '[0 ; ohm*Te/(2*mf);(M-(ohm/2))*Te/mc]';
 B1 = '[L ; f*Poa/mf ; (1-f)*Poa/mc]';
Gc=0.5;
Pd=1.1;
 X1=[1 678.6607 316.9396]';
 Xe=[0 0 0 0 0]';
 E=0;
 DE=0;
 Ync=0;
 Bc=Gc*B;
 Ac=A-Bc*C;
 Xnr=[1];
 Xrr=[2.704e-8];
 to=0;
 h=0.01;
 numofpoints=4000;
 Zr=0;
 T=to;
 for r=1:6,
  t=to+r*h;
  X1=integbdf(eval(A1),X1,eval(B1)*Xnr(r)+eval(BTe),h,r);
  Q2=(1/V)*(((af*(X1(2,r)-X1(2,1))+ac*(X1(3,r)-X1(3,1)))+Xrr(r))*Xnr(r))+(Be/V)*X1(1,r);
  Xnr=integbdf(-Be/V,Xnr,Q2,h,r);
% calculation of error and change in error
  E=[E (Xnr(r)-(1.0+(Pd-1)*((sign(t-2.5)+1)/2)))]
  DE=[DE E(r-1)-E(r)];

           Ync=[Ync fnnout(E(r),DE(r),Mnc,Wnc)];

  Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2)- Xnr(r)+Ync(r+1))

% rod speed limitation
%  Q3=max(-0.0125*Gr,min(Q3,0.0125*Gr));

  Zr=[Zr Q3/Gr];
  Xrr=integbdf(0,Xrr,Q3,h,r);
  Xe=integbdf(Ac-H*C,Xe,Bc*(Xnr(r)-1-Q3/(Gc*Gr))+H*(Xnr(r)-1),h,r);

  T=[T t];
 end

i=8;
t=to+7*h;

while(i<numofpoints)

  X1=integ6bd(eval(A1),X1,eval(B1)*Xnr(i-1)+eval(BTe),h);
  Q2=(1/V)*(((af*(X1(2,i-1)-X1(2,1))+ac*(X1(3,i-1)-X1(3,1)))+Xrr(i-1))*Xnr(i-1))+(Be/V)*X1(1,i-1);
  Xnr=integ6bd(-Be/V,Xnr,Q2,h);
% calculation of error and change in error
   E=[E (Xnr(i-1)-(1.0+(Pd-1)*((sign(t-2.5)+1)/2)))];
  DE=[DE E(i)-E(i-1)];

           Ync=[Ync fnnout(E(i-1),DE(i-1),Mnc,Wnc)];

  Q3=Gr*Gc*( 1.0+(Pd-1)*((sign(t-2.5)+1)/2)- Xnr(i-1)+Ync(i));
 Xnr(i-1)



% rod speed limitation
%  Q3=max(-0.0125*Gr,min(Q3,0.0125*Gr));

  Zr=[Zr Q3/Gr];
  Xrr=integ6bd(0,Xrr,Q3,h);
  Xe=integ6bd(Ac-H*C,Xe,Bc*(Xnr(i-1)-1+Q3/(Gr*Gc))+H*(Xnr(i-1)-1),h);

  T=[T t];
 t=t+h;
 i=i+1;
end

function  [X,Xresidua,Z,U,S,U11,U12,U21,U22]=are_sch(K,R,Q)

%
%      Algebric Riccati Equation Solver [ARE].
%      This program can be used for only time invariant ARE's.
%
%         T
%       K X  +  X  K - X  R  X +  Q  = 0  ;  (A.R.E.)
%
%      Written by F.Erol SAGIROGLU (1991)  Hacettepe U. EEE
%
          [Z]=[ K  -R;-Q  -K']
          [Uo,S]=schur(Z)
          [U]=inv(Uo');
          [n,n]=size(S);
          n=n/2;
          for k=1:n,
           for l=1:n,
            U11(l,k)=U(l,k);
            U12(l,k)=U(l,k-n);
            U21(l,k)=U(l+n,k);
            U22(l,k)=U(l+n,k+n);
           end
          end

   X=U21*inv(U11)

   Xresidua= (K'*X) + (X*K) - (X*R*X) + Q

 % X=inv(U11)'*U21';
  Xresidua= (K'*X) + (X*K) - (X*R*X) + Q;

end

function [X,Xresidua]=are2(L,K,R,Q)

%      Algebraic Riccati Equation Solver
%      The Form of Algebraic Riccati Equation(ARE)
%      Q + L X + X K - X R X = 0 ;
%      Algebraic Riccati equation is reduced into Sylvester Equation
%      Using Newton Iteration Method
%      Written by F.Erol SAGIROGLU (1991) Hacettepe U. EEE

            [n,m]=size(Q);
            Xi=rand(n,m);
 while(1>0),
            [B]=K-R*Xi;
            [A]=L-Xi*R;
            [C]=-(Q+Xi*R*Xi);

%      Sylvester Equation solver;
%      Calculation the upper Hessenberg H and quasi-upper triangular S;
%      H upper Hessenberg,U orthogonal;
          [U,H]=hess(A);
```

```
%      S quasi upper triangular,V orthogonal;
          [V,S]=schur(B);
          [ F]=U'*C*V;
%      So far, transformation  H Y + Y S'= F is completed;
%      So that,solution X = U Y V'


%      Decomposition of A X + X B = C;
          for i=1:m,
           for k=1:n,
            for j=1:m,
             E1((i-1)*n+k,(j-1)*n+k)=H(i,j);
            end
           end
          end
          for i=1:m,
           for k=1:n,
            for j=1:n,
             E2((i-1)*n+k,(i-1)*n+j)=S(j,k);
            end
           end
          end
%      Transformation of F[m,n] into F[n*m,1]
          for i=1:m,
           for j=1:n,
            F1((i-1)*n+j,1)=F(i,j);
           end
          end
           [Y1]=inv(E1+E2)*F1;
           for i=1:m,
            for j=1:n,
             Y(i,j)=Y1((i-1)*n+j,1);
            end
           end
           [X]=U*Y*V';
           E1=E1-E1;E2=E2-E2;
           X
% Error Difference is set to 1e-6
          if max(max(abs(X-Xi)))>1e-6,
            Xi=X;
           disp('new appr..');
          Xresidua=(L*X)+(K*X)+Q-(X*R*X);
        disp(max(max(abs(Xresidua))));
  else
        Xresidua=(L*X)+(K*X)+Q-(X*R*X)
        return;
           end
  end


function [y]=fnnout(E,DE,M,W)

  [n,m]=size(M);
  v=[E;DE];

%Propagation the signals forward through the network using;
   for k=1:n-1,
          H(1:M(k+1),k) = 1.+W(1:M(k+1), sum(M(1:k-1))+1 : sum(M(1:k))) * v ;
```

```
            V(1:M(k+1),k) = tanh( H(1:M(k+1),k) );
            v=V(1:M(k+1),k);
   end;
           y=v;
end;

function [Y]=fuzzyout(E,DE,P)

 for k=1:size(DE,1),

   B=1;
   for i=1:5,
    for j=1:5,
     B=[B mship_1(E(k),i)*mship_2(DE(k),j)];
    end
   end

   B=B(2:size(B,2))/(B(2:size(B,2))*ones(size(B,2)-1,1));

   X=[B*E(k) B*DE(k)];

   Y(k)=X*P;

end

function [P,X,Y,Fout,E,DE]=fuzzyr_1(Yc,Yd,Yp,X)
% Fuzzy Rule Identification using Least Square
% Pseudo-inverse technique
% Yc: controller output as supervisory signal
% Yp: Plant output or plant response to control signal
% Yd:set point, desired plant output

% E:error
E=Yp-Yd;
  DE=E(2:size(E,1))-E(1:size(E,1)-1);
  figure;
  plot(DE,E(1:length(DE)));
 X=zeros(50,50);
 Y=zeros(50,1);
 for k=1:size(DE,1),
   B=1;
   for i=1:5,
    for j=1:5,
     B=[B mship_1(E(k),i)*mship_1(DE(k),j)];
    end
   end
   B=B(2:size(B,2))/(B(2:size(B,2))*ones(size(B,2)-1,1));
   X1=[B*E(k) B*DE(k)];
   X=X+X1'*X1;
   Y=Y+X1'*Yc(k);
   end
   P=pinv(X)*Y;
    Fout=fuzzyout(E,DE,P);
 figure;
 plot(Fout);
 grid;
end
```

```
function [P,X,Y,Fout,E,DE]=fuzzyr_2(Yc,Yd,Yp)
% Fuzzy Rule Identification using Least Square
% Pseudo-inverse technique
% Yc: controller output as supervisory signal
% Yp: Plant output or plant response to control signal
% Yd:set point, desired plant output

% E:error
 E=Yp-Yd;
 DE=E(2:size(E,1))-E(1:size(E,1)-1);
 figure;
 plot(DE,E(1:length(DE)));
X=zeros(50,50);
Y=zeros(50,1);
for k=1:size(DE,1),
  B=1;
  for i=1:5,
   for j=1:5,
    B=[B mship_1(E(k),i)*mship_2(DE(k),j)];
   end
  end
  B=B(2:size(B,2))/(B(2:size(B,2))*ones(size(B,2)-1,1));
  X1=[B*E(k) B*DE(k)];
  X=X-X1'*X1;
  Y=Y-X1'*Yc(k);
  end
  P=pinv(X)*Y;
   Fout=fuzzyout(E,DE,P);
 figure;
 plot(Fout);
 grid;
end

function [X]=Integ6BD(A,X,Q,h)
 % Sixth order Backward Difference Formulas for Integration
 %
 %    X = A X + Q
 %
 % This function is used in simulation of non-linear systems
 %
 % notice: matrix A must be square with the proper sizes of vectors X and Q
 % h refers to step size of integration

 [m,k]=size(X);

    X=[X        -inv(-(147/60)*eye(m)+h*A)*(6*X(:,k)+(-15/2)*X(:,k-1)+(20/3)*X(:,k-2)-(15/4)*X(:,k-
3)+(6/5)*X(:,k-4)-(1/6)*X(:,k-5)+h*Q)];


function [X]=integBDF(A,X,Q,h,r)

  Rk=[ 1    0    0   0    0  0;
     -1/2  2    0   0    0  0;
     1/3  -3/2   3   0    0  0;
     -1/4  4/3  -3   4    0  0;
     1/5  -5/4  10/3  -5    5  0;
     -1/6  6/5 -15/4 20/3 -15/2  6];
  Ck=[-1 -3/2 -11/6 -25/12 -137/60 -147/60];

  [m,k]=size(X);
  X=[X -inv(Ck(r)*eye(m)+h*A)*( X(:,k-r+1:k)*Rk(r,1:r)'+ h*Q)];
function [y]=mship_1(x,i)
% Fuzzy Membership Function(proposed J.Dombi)
% for Servosytem Control
%
boundary=[-1     -0.008 -0.002;
           -0.008 -0.002  0.0  ;
           -0.002  0.0    0.002;
            0.0    0.002  0.008;
            0.002  0.008  1.0  ];
% inflection point of S-shaped membership function
 v=2;
% sharpness of membership function
 l=1;


if (i==1 & x<boundary(1,2)) | (i==5 & x>boundary(5,2))
  y=1;
  return;
end;
if x<boundary(i,1) | x>boundary(i,3)
  y=0;
  return;
end;
if x>=boundary(i,1) & x<boundary(i,2)
  y=((1-v)^(l-1)*(x-boundary(i,1))^l)/((1-v)^(l-1)*(x-boundary(i,1))^l+v^(l-1)*(boundary(i,2)-x)^l);
  return;
end;
if x>=boundary(i,2) & x<=boundary(i,3)
  y=((1-v)^(l-1)*(boundary(i,3)-x)^l)/((1-v)^(l-1)*(boundary(i,3)-x)^l+v^(l-1)*(x-boundary(i,2))^l);
  return;
end;


function [y]=mship_2(x,i)
% Fuzzy Membership Function(proposed by J.Dombi)
% for Servosytem Control
%
boundary=[-1     -0.008 -0.002;
           -0.008 -0.002  0.0  ;
           -0.002  0.0    0.002;
            0.0    0.002  0.008;
            0.002  0.08   1.0  ];
% inflection point of S-shaped membership function
 v=2;
% sharpness of membership function
 l=1;


if (i==1 & x<boundary(1,2)) | (i==5 & x>boundary(5,2))
  y=1;
  return;
end;
if x<boundary(i,1) | x>boundary(i,3)
  y=0;
  return;
end;
if x>=boundary(i,1) & x<boundary(i,2)
  y=((1-v)^(l-1)*(x-boundary(i,1))^l)/((1-v)^(l-1)*(x-boundary(i,1))^l+v^(l-1)*(boundary(i,2)-x)^l);
  return;
end;
if x>=boundary(i,2) & x<=boundary(i,3)
  y=((1-v)^(l-1)*(boundary(i,3)-x)^l)/((1-v)^(l-1)*(boundary(i,3)-x)^l+v^(l-1)*(x-boundary(i,2))^l);
  return;
end;


function [y]=tanh(x)
%sigmoid function used in Neural Networks

   y=(exp(x)-exp(-x))./(exp(x)+exp(-x));


MLNNBP.m
%Multi-Layer Neural Networks Back-Propagation Algorithm-1

   disp('Multi-Layer Neural Network Simulation:');
   M=input('enter neural network definition vector :');
   eta=input('enter the learning factor eta :');

   [n,m]=size(M);

%Step1:
%Initialization of the weights to small random numbers;

   rand('normal');
   Jbp=0;
%    W=rand( max( M(2:n) ), sum( M(1:n-1) ) );

   N=size(Pi,2);
clc;
J=10;
while(J>0.0001)
home
 for i=1:N,

%Step2:
%Choosing a pattern and applying it to the input layer

   v=Pi(1:M(1),i);

%Step3:
%Propagation the signals forward through the network using;
   for k=1:n-1,

          H(1:M(k+1),k) = 1.+W(1:M(k+1), sum(M(1:k-1))+1 : sum(M(1:k))) * v ;

          V(1:M(k+1),k) = tanh( H(1:M(k+1),k) );

          v=V(1:M(k+1),k);

   end;
```

```
%Step4:
%Computation of the deltas for the output layer;

   d=(1. - (V(1:M(n),n-1)).^2).*(Po(1:M(n),i) - V(1:M(k+1),k))
   D(1:M(n),n-1)=d;
%Step5:
%Computation the deltas for the preceeding layers by propagating the error backwards;

   for k=n-1:-1:2,
    D(1:M(k),k-1)=(1. -(V(1:M(k),k-1)).^2).*(W(1:M(k+1), sum(M(1:k-1))+1 : sum(M(1:k)))'*d);
    d=D(1:M(k),k-1);
   end;
%Step6:
%Updating weighting matrix:

  dW(1:M(2), 1:M(1) )=eta.*(D(1:M(2),1)*Pi(1:M(1),i)');

  for k=2:n-1,
   dW(1:M(k+1), sum(M(1:k-1))+1:sum(M(1:k)))=eta.*(D(1:M(k+1),k)*V(1:M(k),k-1)');
  end;

  W = W + dW;
%Step7:
%Repeat for the next pattern:

 end;


%Calculating cost function:
 J1=J;
 J=0;
 Pout=1;
 for i=1:N,

   v=Pi(1:M(1),i);

%Propagation the signals forward through the network using;
   for k=1:n-1,

          H(1:M(k+1),k) = 1.+W(1:M(k+1), sum(M(1:k-1))+1 : sum(M(1:k))) * v ;

          V(1:M(k+1),k) = tanh( H(1:M(k+1),k) );

          v=V(1:M(k+1),k);

   end;

%Step4:
%Computation sum of squares as cost function
   Pout=[Pout v];
   J=J+(Po(1:M(n),i) - V(1:M(k+1),k))^2;
 end;
 J=sqrt(J)/N;
 figure(1);
 plot(Pi,Pout(2:N+1),Pi,Po,'x');
 xlabel(num2str(J));
text(1,0,num2str(eta))
if J>J1,
    eta=9*eta/10;
  end
  if J<J1,
  %    eta=10.1*eta/10;
  end
  Jbp=[Jbp J];
 end


%MLNNKF.m
% Learning Algorithm via Extended Kalman Filter
% for MultiLayered Feedforward Neural Networks

   disp('Multi-Layer Neural Network Simulation:');
   M=input('enter neural network definition vector :');

   [n,m]=size(M);

%Step1:
%Initialization of the weights to small random numbers:

   rand('normal');


  W=rand( max( M(2:n) ), sum( M(1:n-1) ) );
 %save W;
  Wkf=W;

%error covariance of pseudo-noise to tune weights
   R=ones( max( M(2:n) ), sum( M(1:n-1) ) );

%Error covariance initial matrix
  P=1*ones( max( M(2:n) ), sum( M(1:n-1) ) );

   N=size(Pi,2);
%clc;
%Jkf=1;
%J=10;
home
while(J>0.0001)

  for i=1:N,

%Step2:
%Choosing a pattern and applying it to the input layer

   v=Pi(1:M(1),i);

%Step3:
%Propagation the signals forward through the network using:
   for k=1:n-1,

         H(1:M(k+1),k) = 1.+W(1:M(k+1), sum(M(1:k-1))+1 : sum(M(1:k))) * v ;

         V(1:M(k+1),k) = tanh( H(1:M(k+1),k) );

         v=V(1:M(k+1),k);
    end;

 %Step4:
 %Computation of the deltas for the output layer;

    d=(1. - (V(1:M(n),n-1)).^2).*(Po(1:M(n),i) - V(1:M(k+1),k))
    D(1:M(n),n-1)=d;
 %Step5:
 %Computation the deltas for the preceeding layers by propagating the error backwards;

    for k=n-1:-1:2,
     D(1:M(k),k-1)=(1. -(V(1:M(k),k-1)).^2).*(W(1:M(k+1), sum(M(1:k-1))~1 : sum(M(1:k)))'*d);
     d=D(1:M(k),k-1);
    end;
 %Step6:
 %Updating weighting matrix;

   dW(1:M(2), 1:M(1) )=(D(1:M(2),1)*Pi(1:M(1),i)');

   for k=2:n-1,
    dW(1:M(k+1), sum(M(1:k-1))+1:sum(M(1:k)))=(D(1:M(k+1),k)*V(1:M(k),k-1)');
   end;

                K=(P.*dW*(1/(Po(1:M(n),i)      -      V(1:M(k+1),k)))./(P.*(dW*(1/(Po(1:M(n),i)      -
 V(1:M(k+1),k)))).^2+R);
   W = W - K*(Po(1:M(n),i) - V(1:M(k+1),k));

   P=(1.-K.*dW*(1 (Po(1:M(n),i) - V(1:M(k+1),k))) ).*P;

 %Step7:
 %Repeat for the next pattern;

  end;

   Wkf=[Wkf W];



 %Calculating cost function;
  J=0;
  Pout=1;
  for i=1:N,

   v=Pi(1:M(1),i);

 %Propagation the signals forward through the network using;
   for k=1:n-1,

           H(1:M(k+1),k) = 1.+W(1:M(k+1), sum(M(1:k-1))+1 : sum(M(1:k))) * v ;

           V(1:M(k+1),k) = tanh( H(1:M(k+1),k) );

           v=V(1:M(k+1),k);

   end;

 %Step4:
 %Computation sum of squares as cost function
```

```
    Pout=[Pout v];
    J=J+(Po(1:M(n),i) - V(1:M(k+1),k))^2;
   end;
   J=sqrt(J)/N;
   figure(1);
  % plot(Pi,Pout(2:N+1),Pi,Po,'x');
    plot(1:N,Pout(2:N+1),1:N,Po);
  xlabel(num2str(J));
   Jkf=[Jkf J];
 end


%MLKFBDF.m
%Multi-Layered Feedforward Neural Networks
%Extended Kalman Filter Learning Algorithm-1
%Modified with Backward Difference Formulas

   disp('Multi-Layer Neural Network Simulation:');
   M=input('enter neural network definition vector :');

   [n,m]=size(M);

%Step1:
%Initialization of the weights to small random numbers;

   rand('normal');

   W=rand( max( M(2:n) ), sum( M(1:n-1) ) );
    [nW,mW]=size(W);
   Wx=reshape(W,nW*mW,1);

%error covariance of pseudo-noise to tune weights
   R=0.1*ones( max( M(2:n) ), sum( M(1:n-1) ) );

%Error covariance initial matrix
   P=1*ones( max( M(2:n) ), sum( M(1:n-1) ) );

   N=size(Pi,2);
clc;
J=10;
home

 r=1;
while(J>0.001),

  for i=1:N,

%Step2:
%Choosing a pattern and applying it to the input layer

   v=Pi(1:M(1),i);

%Step3:
%Propagation the signals forward through the network using;
   for k=1:n-1,

          H(1:M(k+1),k) = 1.+W(1:M(k+1), sum(M(1:k-1))+1 : sum(M(1:k))) * v ;
          V(1:M(k+1),k) = tanh( H(1:M(k+1),k) );

          v=V(1:M(k+1),k);

   end;

%Step4:
%Computation of the deltas for the output layer;

   d=(1. - (V(1:M(n),n-1)).^2).*(Po(1:M(n),i) - V(1:M(k+1),k))
   D(1:M(n),n-1)=d;
%Step5:
%Computation the deltas for the preceeding layers by propagating the error backwards;

   for k=n-1:-1:2,
     D(1:M(k),k-1)=(1. -(V(1:M(k),k-1)).^2).*(W(1:M(k+1), sum(M(1:k-1))+1 : sum(M(1:k)))'*d);
     d=D(1:M(k),k-1);
   end;
%Step6:
%Updating weighting matrix:

  dW(1:M(2), 1:M(1) )=(D(1:M(2),1)*Pi(1:M(1),i)');

  for k=2:n-1,
    dW(1:M(k+1), sum(M(1:k-1))+1:sum(M(1:k)))=(D(1:M(k+1),k)*V(1:M(k),k-1)');
  end;

                K=(P.*dW*(1/(Po(1:M(n),i)      -      V(1:M(k+1),k)))).(P.*(dW*(1/(Po(1:M(n),i)      -
V(1:M(k+1),k)))).^2+R);
  dW1 = K*(Po(1:M(n),i) - V(1:M(k+1),k));

  P=(1.-K.*dW*(1/ (Po(1:M(n),i) - V(1:M(k+1),k))) ).*P;

% Integration using BDF

   dWx=reshape(dW1,nW*mW,1);

  if r<7,
    Wx=integbdf(0,Wx,dWx,1,r);
    r=r+1;
  else
    Wx=integ6bd(0,Wx,dWx,1);
  end;

   W=reshape(Wx(:,size(Wx,2)),nW,mW);
%   W=W+dW1;

%Step7;
%Repeat for the next pattern;

  end;



%Calculating cost function;
  J=0;
  Pout=1;
```

```
 for i=1:N,

   v=Pi(1:M(1),i);

%Propagation the signals forward through the network using;
   for k=1:n-1,

          H(1:M(k+1),k) = 1.+W(1:M(k+1), sum(M(1:k-1))+1 : sum(M(1:k))) * v ;

          V(1:M(k+1),k) = tanh( H(1:M(k+1),k) );

          v=V(1:M(k+1),k);

   end;

%Step4:
%Computation sum of squares as cost function
   Pout=[Pout v];
   J=J+(Po(1:M(n),i) - V(1:M(k+1),k))^2;
 end;
 J=sqrt(J)/N;
 figure(1);
 plot(Pi,Pout(2:N+1),Pi,Po,'x');
 xlabel(num2str(J));

 end

% EXPBDF.m
% System Simulation Test using
% Backward Difference Formulas (BDFs) in Integration
% for exponential inputs

B= '[0]';
a=1000000;
Q='[exp(-a*t)]';

X(:,1)=[0]';
to=0;
h=0.00001;
m=1;
numofpoints=200;

 k=1;
 Erms=0;
 H=0;
while(h<=0.0001)

 t=to+h;
X(:,2)=-inv(-eye(m)+h*eval(B))*(X(:,1)+h*eval(Q));

t=to+2*h;
X(:,3)=-inv(-(3/2)*eye(m)+h*eval(B))*(2*X(:,2)-(1/2)*X(:,1)+h*eval(Q));

t=to+3*h;
X(:,4)=-inv(-(11/6)*eye(m)+h*eval(B))*(3*X(:,3)-(3/2)*X(:,2)+(1/3)*X(:,1)+h*eval(Q));

t=to+4*h;
X(:,5)=-inv(-(25/12)*eye(m)+h*eval(B))*(4*X(:,4)-3*X(:,3)+(4/3)*X(:,2)-(1/4)*X(:,1)+h*eval(Q));
```

```
t=to+5*h;
X(:,6)=-inv(-(137/60)*eye(m)+h*eval(B))*(5*X(:,5)-5*X(:,4)+(10/3)*X(:,3)-
(5/4)*X(:,2)+(1/5)*X(:,1)+h*eval(Q));

t=to+6*h;
X(:,7)=-inv(-(147/60)*eye(m)+h*eval(B))*(6*X(:,6)-(15/2)*X(:,5)+(20/3)*X(:,4)-
(15/4)*X(:,3)+(6/5)*X(:,2)-(1/6)*X(:,1)+h*eval(Q));

i=8;
t=to+7*h;
while(i<numofpoints)
X(:,i)=-inv(-(147/60)*eye(m)+h*eval(B))*(6*X(:,i-1)-(15/2)*X(:,i-2)+(20/3)*X(:,i-3)-(15/4)*X(:,i-
4)+(6/5)*X(:,i-5)-(1/6)*X(:,i-6)+h*eval(Q));
t=t+h;
i=i+1;
end
figure;
plot(X);
title(num2str(h));
grid;
 Xa=0;
 T=0;
 E=0;
for i=1:(numofpoints-1),
 Xa(i)=(1/a)*(1-exp(-a*(i*h-h)));
 T(i)=i*h-h;
 E=E+(X(i)-Xa(i))^2;
end
 Erms(k)=sqrt(E/numofpoints);
 H(k)=h;
 k=k+1
 h=h+0.00001;
end
clg;
plot(H,Erms);
xlabel('step size');
ylabel('error rms');
end.

% SINBDF.m
% System Simulation Test using
% Backward Difference Formulas (BDFs) in Integration

B= '[0]';

Q='[sin(100*t)]';

X(:,1)=[-1]';
to=0;
h=0.001;
m=1;
numofpoints=500;

 k=1;
 Erms=0;
 H=0;
while(h<=0.01)
```

```
t=to+h;
X(:,2)=-inv(-eye(m)+h*eval(B))*(X(:,1)+h*eval(Q));

t=to+2*h;
X(:,3)=-inv(-(3/2)*eye(m)+h*eval(B))*(2*X(:,2)-(1/2)*X(:,1)+h*eval(Q));

t=to+3*h;
X(:,4)=-inv(-(11/6)*eye(m)+h*eval(B))*(3*X(:,3)-(3/2)*X(:,2)+(1/3)*X(:,1)+h*eval(Q));

t=to+4*h;
X(:,5)=-inv(-(25/12)*eye(m)+h*eval(B))*(4*X(:,4)-3*X(:,3)+(4/3)*X(:,2)-(1/4)*X(:,1)+h*eval(Q));

t=to+5*h;
X(:,6)=-inv(-(137/60)*eye(m)+h*eval(B))*(5*X(:,5)-5*X(:,4)+(10/3)*X(:,3)-
(5/4)*X(:,2)+(1/5)*X(:,1)+h*eval(Q));

t=to+6*h;
X(:,7)=-inv(-(147/60)*eye(m)+h*eval(B))*(6*X(:,6)-(15/2)*X(:,5)+(20/3)*X(:,4)-
(15/4)*X(:,3)+(6/5)*X(:,2)-(1/6)*X(:,1)+h*eval(Q));

i=8;
t=to+7*h;
while(i<numofpoints)
X(:,i)=-inv(-(147/60)*eye(m)+h*eval(B))*(6*X(:,i-1)-(15/2)*X(:,i-2)+(20/3)*X(:,i-3)-(15/4)*X(:,i-
4)+(6/5)*X(:,i-5)-(1/6)*X(:,i-6)+h*eval(Q));
t=t+h;
i=i+1;
end
figure;
plot(X);
xlabel(num2str(h));
grid
 Xa=0;
 T=0;
 E=0;
for i=1:(numofpoints-1),
 Xa(i)=-cos(i*h-h);
 T(i)=i*h-h;
 E=E+(X(i)-Xa(i))^2;
end
 Erms(k)=sqrt(E/numofpoints);
 H(k)=h;
 k=k+1
 h=h+0.001;
end
clg;
plot(H,Erms);
xlabel('step size');
ylabel('error rms');
end.
```

## AUTOBIOGRAPHY

He was born in Ankara on October 29, 1969. He received the BSEE with honors from the electrical and electronics engineering department of Hacettepe University, Ankara. He is also entitled to get the Encouragement Award of Student's Scientific Researches by Hacettepe University, in 1991.

The emphasis of his postgraduate work is on the mutidisciplinary studies in electronics, optics and mechanics in academy and industry. In 1991-1992, he was graduate research assistant in the laser-optics laboratory and participant of a defense project in Bilkent University, Ankara. During 1992-1996, he was graduate student in the Nuclear Energy Institute, Istanbul and employed by Evre Electronics Inc., Istanbul and Meksan Industries Inc., Istanbul as R&D engineer and electronic engineer. He has done research and development and design in Riccati differential equations, fuzzy systems, neural networks, digital controllers for electric motors and Arcelik automatic washing mashines and the automation of the fuel pumping systems. He is currently in the national service in the army.