ABSTRACT

This thesis investigates the use of fuzzy logic and neural networks in the design of power system stabilisers.

Power system stabilisers are used to enhance the damping of low frequency oscillations in the shaft speed of the generators used in a power system. The most widely used power system stabiliser is a lead-lag compensator known as a conventional power system stabiliser. The design of conventional power system stabilisers is based on linearised model of the power system for a specific operating point. The gain settings of the conventional power system stabiliser are fixed. Thus, the conventional power system stabiliser has a nearly optimum response for the specific operating condition for which it was designed. However, operating conditions of a power system change as a result of load changes or unpredictable disturbances. The performance of the power system is especially important when major disturbances such as faults occur in the system. The main disadvantage of the conventional power system stabiliser is that it cannot adapt to changes in operating conditions.

In recent years fuzzy logic power system stabilisers have been considered as better alternatives to conventional power system stabilisers. Because of their capability of handling uncertainties they are generally more robust than conventional power system stabilisers. Compared with conventional power system stabilisers, fuzzy logic power system stabilisers have better performance over a wide range of operating conditions.

In spite of the aforementioned fact about fuzzy logic power system stabilisers, their performance depends on the operating conditions, too. The performance of the fuzzy logic power system stabilisers with fixed parameters will be optimum for a specific operating conditions and will degrade for other operating conditions. However, this degradation is less compared to the conventional power system stabiliser.

In order to achieve an optimum response, two strategies have been followed in this

thesis:

In the first strategy, an attempt has been made to tune the fuzzy logic power system stabiliser while the generator is operating on-line. Two schemes have been used. In the first scheme a feedforward artificial neural network has been employed to tune the fuzzy power system stabiliser. In the second scheme a fuzzy logic system has been used to do the tuning.

In the second strategy, adaptive power system stabilisers have been designed using artificial neural networks and fuzzy logic:

Firstly, a hierarchical artificial neural network has been proposed as an adaptive neural network power system stabiliser. The architecture of the hierarchical neural network consists of two sub-networks. One sub-network is used for the system identification of the nonlinear power plant involved and the other one is used as a stabiliser. The weights of the neural network stabiliser are adjusted according to the difference between the output of the neural network identifier and a desired output. The neural network stabiliser and neural network identifier are trained in different stages by the backpropagation algorithm.

Secondly, two adaptive fuzzy logic power system stabilisers have been developed using the concept of fuzzy basis functions. In the first scheme, which is known as an indirect adaptive scheme, the power system is modelled by differential equations with nonlinear parameters which are functions of the state of the system. These nonlinear functions may not be known, except that some linguistic information is available about them. Utilising this information, fuzzy logic systems are designed to model the system behaviour. The control law is obtained using the uncertainty principle. Based on the Lyapunov's synthesis method, adaptation rules are developed to make the controller adaptive to changes in operating conditions of the power system. In the second scheme, a direct adaptive fuzzy logic power system stabiliser have been developed. The linguistic rules, regarding the dependence of the plant output on the controlling signal, have been used to build the initial fuzzy logic power system stabiliser. This is different with the first

scheme because the linguistic rules are directly used to construct the adaptive fuzzy logic power system stabiliser. Based on the Lyapunov's direct method, an adaptation rule is developed in order to make the fuzzy logic power system stabiliser adapt to the changes in operating conditions of the power system.

Finally the responses of the power system for various power system stabilisers proposed in the thesis have been obtained using nonlinear simulations. Their performances have been compared for various operating conditions and system configurations. A performance index has been defined to conduct quantitative comparisons. Concluding remarks have been given on the basis of the results obtained. Finally, some suggestions have been given for further research.

ACKNOWLEDGEMENTS

I would like to thank my supervisor **Professor Akhtar Kalam**, the Director of Save Energy Research Group and the Head of the Department of Electrical and Electronic Engineering in Victoria University of Technology, for his support, invaluable comments and encouragements throughout this work. Special appreciation goes to my co-supervisor **Dr. Wee Sit Lee** for his constructive comments and suggestions.

I greatly enjoyed inspiration and friendly environment provided by the staff and postgraduate students of the Department of Electrical and Electronic Engineering of VUT. My special thanks go to my colleagues: **Mahmood**, **Reza**, **Mehrdad**, **Omar**, **Rushan**, **Zahidul**, **Ranjan** and **Navaratnam**.

I would like to acknowledge the **Ministry of Culture and Higher Education** of the **Islamic Republic of Iran** who financially supported me during the research.

Last, but not least, I wish to express my gratitude for the support and encouragement I have received from my family. I am specially indebted to my wife **Fatemeh** for her great support, to my son **Mohammad**, and my daughters **Zahra** and **Maryam** who made it all possible. I would also like to appreciate the support of **my parents, brothers and sisters** during my studies.

ACRONYMS

AFPSS	Adaptive Fuzzy Logic Power System Stabiliser
ANN	Artificial Neural Network
APSS	Adaptive Power System Stabiliser
ANNPSS	Adaptive Neural-Network-Based Power System Stabiliser
AVR	Automatic Voltage Regulator
CPSS	Conventional (Linear) Power System Stabiliser
DAFPSS	Direct Adaptive Fuzzy Logic Power System Stabiliser
DOF	Degree of Firing
FBF	Fuzzy Basis Function
FLC	Fuzzy Logic Controller
FLS	Fuzzy Logic System
FNN	Feedforward Neural Network
FPSS	Fuzzy Logic Power System Stabiliser
IAFPSS	Indirect Adaptive Fuzzy Logic Power System Stabiliser
NFPSS	Fuzzy Logic Power System Stabiliser tuned by a Neural Network
PSS	Power System Stabiliser
TFPSS	Fuzzy Logic Power System Stabiliser tuned by a FLS

NOTATIONS

General notations:

J_p	Performance Index of the PSS
K_p	Speed scaling factor
K_d	Acceleration scaling factor

Fuzzy logic notations:

$\rightarrow$	Fuzzy implication operator
$\star$	Triangular norm (T-norm) operator
o	Composition operator
$\oplus$	Union operator
$\mu_A(x)$	Membership degree of the numerical variable x in the fuzzy set A

Power system notations:

$\tilde{E}_B$	The infinite bus voltage as a phasor
$\tilde{E}'$	Internal voltage of a generator behind the transient reactance as a phasor
$\tilde{E}''$	Internal voltage of a generator behind the subtransient reactance as a phasor
P_e	The air-gap electrical power of a generator
P_t	Terminal active power of a generator
Q_t	Terminal reactive power of a generator
R_E	Thevenin resistance of the network
T_m	Mechanical torque applied to a generator
T_e	The air-gap torque of a generator
$\tilde{V}_t$	Terminal voltage of a generator as a phasor
X_E	Thevenin reactance of the network
X_{tr}	Reactance of the transformer
δ	Rotor angle

ω_r	Angular speed of the shaft
$\Delta\omega_r$	Speed deviation with respect to the synchronous speed

Notations representing the parameters of a synchronous machine:

E_{fd}	Exciter output voltage
i_{fd}	Rotor (field) circuit current
L_{ads}	Saturated d-axis mutual inductance between stator and rotor windings
L_{adu}	unsaturated direct-axis mutual inductance between stator and rotor windings
L_{aqs}	saturated q-axis mutual inductance between stator and rotor windings
L_{fd}	Self-Inductance of the field circuit
L_l	Leakage inductance of the stator
R_a	Armature resistance
R_{fd}	Rotor (field) circuit resistance
X_d'	Direct-Axis transient reactance of a synchronous machine
X_d''	Direct-Axis subtransient reactance of a synchronous machine
ψ_{fd}	Rotor circuit (field) flux linkage

CONTENTS

LIST OF FIGURES

LIST OF TABLES

PUBLICATIONS

The following is a list of publications from the work presented in the thesis.

a) Journal papers:

[1] Nasser Hosseinzadeh and Akhtar Kalam, "A rule-based fuzzy power system stabilizer tuned by a neural network", Accepted to be Published in the IEEE Transactions on Energy Conversion.

[2] Nasser Hosseinzadeh and Akhtar Kalam, "A hierarchical neural network adaptive power system stabiliser", Accepted to be Published in the International Journal of Power and Energy Systems, vol. 18, no. 1, 1998.

[3] Nasser Hosseinzadeh and Akhtar Kalam, "A direct adaptive fuzzy power system stabiliser", Accepted to be Published in the IEEE Transactions on Energy Conversion, PE-069-EC-0-12-1997.

b) Conference papers:

[1] Nasser Hosseinzadeh and Akhtar Kalam, "On-Line tuning of a fuzzy power system stabiliser", International Conference on Energy Management and Power Delivery, IEEE Catalogue No. 98EX137, Singapore, 3-5 March 1998.

[2] Nasser Hosseinzadeh and Akhtar Kalam, "A direct adaptive fuzzy power system stabiliser", Fourth International Conference on Advances in Power System Control, Operation and Management (APSCOM-97), Hong Kong, November 11-14 1997, pp. 377-382.

[3] Nasser Hosseinzadeh and Akhtar Kalam, "An indirect adaptive fuzzy power system stabiliser", Australian Universities Power Engineering Conference (AUPEC'97) and Electric Energy Conference (EECON'97), Sydney, Australia, 29 Sept.-1 Oct. 1997, vol. 2, pp. 253-258.

[4] Nasser Hosseinzadeh, Akhtar Kalam and Wee Sit Lee, "An adaptive power system stabiliser using neural networks", Australian Universities Power Engineering Conference (AUPEC'96), Melbourne, Australia, October 2-4 1996, pp. 605-609.

[5] Nasser Hosseinzadeh and Akhtar Kalam, "A neuro-fuzzy power system stabiliser", The 22nd Annual International Conference of the IEEE Industrial Electronics Society (IECON'96), Taipei, Taiwan, August 5-9 1996, pp. 608-613

[6] Nasser Hosseinzadeh, Akhtar Kalam and Wee Sit Lee, "Tuning a fuzzy logic power system stabiliser using neural networks", The First International Discourse on Fuzzy Logic and the Management of Complexity, Sydney, Australia, January 15-18 1996, pp. 201-205.

[7] Nasser Hosseinzadeh, Akhtar Kalam and Wee Sit Lee, "A comparison study between a conventional and a fuzzy logic based power system stabiliser for a synchronous generator", 10th International Power System Conference, Tehran, Iran, November 6-8 1995, pp. 183-191.

[8] Nasser Hosseinzadeh, Akhtar Kalam and Wee Sit Lee, "A fuzzy logic based power system stabiliser for a synchronous generator", Electrical Energy Conference, Adelaide, Australia, September 25-27 1995, pp. 176-181.

Chapter 1

Introduction

1.1 Power System Stability

Power system stability may be broadly defined as that property of a power system that enables it to remain in a state of operating equilibrium under normal operating conditions and to regain an acceptable state of equilibrium after being subjected to a disturbance [1]. Since power systems rely on synchronous machines for generation of electrical power, a necessary condition for satisfactory system operation is that all synchronous machines remain in synchronism. This aspect of stability is influenced by the dynamics of generator rotor angles and power-angle relationships.

Electric power systems are highly complicated systems that contain nonlinear and time varying elements. Their dynamics cover a wide spectrum of phenomena, which are electrical, electro-mechanical, electro-magnetic, and thermal in nature. The highly interconnected nature of power systems makes their operation and control a complex process. Thus disturbances in some elements may affect the whole system operation and stability causing poor power quality or even interruption of power

supply [2, 3].

The stability of power systems was first recognized as an important problem in 1920 [4]. Results of the first laboratory tests on miniature systems were reported in 1924 [5]; the first field tests on the stability on a practical power system were conducted in 1926 [6, 7].

Early stability problems were associated with remote hydro-electric generating stations feeding metropolitan load centres over long-distance transmission lines. For economical reasons, such systems were operated close to their steady state stability limits. In a few instances, instability occurred during steady state operation. Instability occurred more frequently following short-circuits and other system disturbances [8]. The stability problem was largely influenced by the strength of the transmission system, with instability being the result of insufficient synchronising torque.

As power systems evolved and interconnections between independent systems were found to be economically attractive, the complexity of the stability problems increased. When the problems of power system instability became more serious, they required the attention of power engineers [9]. Since 1950, extensive research has been conducted to overcome power system stability problems.

For the purpose of analytical studies, researchers have classified power system stability into two categories [10, 11, 12]:

1. Steady state stability (or dynamic stability)

Steady state stability corresponds to the stability of the power system around an operating point. If the system is able to maintain synchronism after small changes has happened in operating conditions, it is said that it has steady state stability.

Typical perturbations under this category may be small, randomly occurring changes in loads or small alterations in reference voltage settings. If the system is stable, it is expected that after a temporary small disturbance the system will return to its initial state. For a permanent small disturbance the system will acquire a new operating point after a transient period [2]. In both cases the synchronism of the system should not be lost. The size of small disturbances may be measured by the criterion that the perturbed system remains in an approximately linear region [2]. These type of disturbances can lead to long term sustained oscillations [10]. In the literature, steady state stability is also referred to as dynamic stability.

2. Transient stability

Transient stability refers to the ability of the power system to maintain stability after a sudden and severe disturbance. System faults, line switching, and large changes in loads can be considered as severe disturbances that lead to transient stability problems. It is usually assumed that the system under study is stable before a large disturbance happens. If the system has transient stability, the system oscillations resulting from large disturbances are damped. However, transient stability of the system depends very much on the initial operating condition of the system and the nature (i.e., the type, magnitude, duration, and location, etc.) of the disturbances that are applied to the system, and on the post-fault system configuration as well [2].

1.2 Power System Control

Many techniques have been proposed to overcome the stability problems as mentioned in Section 1.1. These include load frequency control and governor control [13, 14, 15], capacitor switching control [16, 17], and excitation control [18, 19, 20].

Out of these methods, most attention has been given to excitation control. The reasons for this can be summarised as follows:

1. The synchronous generator excitation loop has a small time constant compared to the governor time constant. Therefore, excitation control gives a faster reaction to disturbances.

2. Excitation control can have a large effect with a relatively small control energy.

3. In practice, it is easier to deal with the control of the electric field circuit rather than the mechanical governor.

4. Excitation control requires less cost compared to other methods.

1.3 Excitation control of power systems

In the last few decades, considerable attention has been given to the excitation system and its role in improving power system stability [12, 21]. By using a voltage regulator in the excitation control system, the voltage profile at the consumer end can be improved with respect to the change of the demand for reactive power. At first, manual voltage regulators were used. Nowadays, all the generating units are equipped with *Automatic Voltage Regulators* (AVRs). Using negative feedback of machine terminal voltage, the AVR applies a control signal to the generator excitation. The AVR can be approximated by a linear first order transfer function as shown in equation (1.1):

$$AVR(s) = \frac{K_a}{(1 + T_a s)} \tag{1.1}$$

Early researchers suggested the use of high-gain AVRs to increase the steady

state power limits of power systems [22, 23]. A high-gain AVR was also recommended for reducing the steady state error of the system output. Gradually it became apparent that the voltage regulating action of the high-gain AVRs had a negative impact upon the steady state stability of power systems. It was observed that oscillations of small magnitude and low frequency often persisted for long periods of time. Insufficient damping of these oscillations may limit the ability of generators to transmit power [18]. In some cases, inappropriate selection of the voltage regulator may greatly decrease the system damping and may even lead to negative damping [14, 24]. However, this did not seem to have serious problems in early implementation [25].

In order to enhance the performance of the AVRs, researchers proposed the use of *power system stabilisers* (PSSs). These stabilisers inject a supplementary control signal to the excitation control loop to improve the damping characteristics of the system [25, 26]. Over the past four decades, many types of PSSs have been extensively studied, and some of them have been successfully used in the industry.

1.4 Conventional Power System Stabilisers

In the literature considerable efforts have been placed on the application of PSSs, which are designed on the basis of conventional linear control techniques. These kinds of PSSs are known as *Conventional Power System Stabilisers* (CPSSs) [28, 29, 30]. The CPSS usually employed by the utility industry is a lead-lag network using the speed deviation as input. The fundamental concept for the design of such a CPSS is to compensate the phase lag resulting from the AVR, the exciter, and the generator so that a supplementary damping torque component in phase with the rotor speed is generated [10]. This supplementary damping torque can be employed to enhance the dynamic stability of the power system.

A CPSS consists of three blocks [2]: a phase compensation block, a signal washout block, and a gain block.

The *phase compensation* block provides the appropriate phase-lead characteristic to compensate for the phase lag between the exciter input and the generator electrical torque. Normally, two first-order blocks are used to achieve the desired phase compensation. More blocks may be used if required, but two blocks are usually sufficient. Typically, the frequency range of interest is 0.1 to 2.0 Hz, and the phase-lead network should provide compensation over this entire frequency range.

The *signal washout* block serves as a high-pass filter, with the time constant high enough to allow signals associated with oscillations in shaft speed to pass unchanged. Without it, small steady changes in speed would modify the terminal voltage. The time constant of the washout block should be long enough to pass stabilising signals at the frequencies of interest unchanged, but not so long that it leads to undesirable generator voltage excursions during system-islanding conditions. These conditions are unusual combination of circumstances and events which cause a portion of the interconnected system to separate completely and form one or more electrical islands.

The *stabiliser gain* determines the amount of damping introduced by the PSS. Ideally, the gain should be set at a value corresponding to maximum damping; however, it is often limited by other considerations.

With these three blocks, the CPSS with one input takes the form presented by equation (1.2):

$$G(s) = K_{pss}\frac{T_w s}{1+T_w s}\frac{(1+T_1 s)(1+T_3 s)}{(1+T_2 s)(1+T_4 s)} \tag{1.2}$$

In this equation the first term is the stabiliser gain, the second term is the signal washout block, and the other two terms are two first-order phase compensation blocks.

Using the speed as an input signal has some disadvantages. The speed signal can be corrupted by torsional oscillations of the shaft. Also, the effective gain of the stabiliser path (machine gain) decreases under weak system conditions when the stabiliser is most needed [19].

Various PSS input signals have been used, e.g. the shaft speed [27], the ac system frequency [26, 31, 32], the accelerating power [33], and a combination of shaft speed and electrical power [34]. The rate of change of the terminal voltage has also been used [35].

The design of CPSSs is based on a linear model of the power system at some operating point. The classical control theory, described in terms of transfer functions, is employed as the design tool for the CPSS [18, 19, 20]. By implementing CPSSs in the generating units, the stability limits of the power systems have improved considerably. In fact these kinds of PSSs have made a great contribution in enhancing the quality of operation of power systems. Their performance at the designed operating point can be excellent. However, it can deteriorate with the change in operating point. This is because the CPSSs are designed using a linearised model of the machine at a prescribed operating point.

In practice, the systems are highly nonlinear. For example, the gain of the plant increases with the generator loading and the transmission line strength [19]. Also, the phase lag of the plant increases as the system becomes stronger (the power system is interconnected with lower impedances transmission lines). Thus, controller parameters which are optimum for one set of operating conditions may not be

suitable for another set of operating conditions. This is the major disadvantage of the CPSS design method, i.e., it does not guarantee system stability under varying operating conditions [36]. This has opened the door for research using modern control techniques.

1.5 Modern Power System Stabilisers

As mentioned in Section 1.4 the nonlinear nature of power systems makes CPSSs inappropriate for varying operating conditions. This nonlinearity raises some difficulties such as: (1) how to effectively tune the PSS parameters, (2) how to track and compensate for the variation of system operating conditions, and (3) how to handle the interactions between the various machines. A lot of research has been conducted to overcome these difficulties. Various PSS transfer functions associated with different systems have been proposed [18, 19]. Different techniques to effectively tune and optimise the CPSS have been studied [37, 38, 40, 41, 42]. Effective placing and coordination between different PSSs in a multi-machine environment has been introduced [43, 44, 45, 46]. Effect of the output limiter on the performance of a PSS has been studied [47]. Augmented PSSs to improve the performance of CPSSs have been proposed [48, 49]. A discrete mode PSS, which is optimised based on the integral of squared error (ISE), has also been proposed [50].

With the increasing demand for quality electric supply, other modern control design techniques were introduced:

Linear optimal control strategy is one of such techniques [38, 39, 44, 51, 52, 53, 54]. The main drawback of linear optimal controllers is that they are also based on a linearised system model. Thus, they face the same limitations as faced by the CPSSs. Recently, optimisation techniques based on genetic algorithms have been used to tune the PSS [55].

Adaptive control was found to solve some of the aforementioned problems. The field of adaptive control was developed to control systems which are time-varying. These variations may be caused by disturbances on the system or changes in the operating conditions. In the past three decades major advances have been made in the identification and control of linear plants with unknown parameters [56].

Adaptive and self-tuning PSSs have been constructed as an alternative to CPSSs [57, 58, 59, 60, 61]. The advantage of the adaptive power system stabilisers (APSSs) is their ability to adjust the controller parameters on-line according to the current operating conditions. Researchers have found that under large variations in operating conditions APSSs perform better than CPSSs [62, 63, 64, 65, 66, 67, 68]. In reference [62] an on-line discrete auto regressive moving average identification strategy has been used to model the dynamics of the power system. Based on the identified model, an APSS is designed by using a self-optimising pole shifting control algorithm. In reference [63] adaptive control algorithms which use least-squares identification with different strategies have been explained. In reference [64] an adaptive proportional-integral (PI) controller has been developed. The gains of the proposed controller have been adjusted in real time using the on-line measured operating conditions and a look-up table stored in computer memory. In reference [65] a multi-input multi-output pole-shifting control algorithm together with a least-square system identification has been used to implement an APSS. It has been tested on a physical model of a power system. Pole-shifting control algorithm has also been used in references [66] and [67] to design APSSs. In reference [68] a discrete-time adaptive sliding mode control has been developed and applied to the PSS problem. A controllable canonical form of state space realisation has been constructed using the parameters identified by the on-line recursive least squares method. All these APSSs are based on adaptive techniques for linear systems.

System identification is necessary to design these APSSs. While major advances have been made in the design of adaptive controllers for linear systems with unknown parameters, such controllers cannot be used for the global control of nonlinear systems. Research is going on in the field of adaptive controllers for nonlinear systems [69].

Robust H_∞ control and μ synthesis have been successfully applied to the design of PSSs [70, 71, 72, 73, 74, 75, 76, 77, 78]. Also robust nonlinear controller has been proposed as a PSS [79, 80]. The H_∞ optimisation and μ synthesis schemes provide a theoretical mechanism to deal with uncertainties in a system. In the design process of PSSs based on H_∞ control and μ synthesis a model with an uncertainty description is used for the power system. The uncertainties are due to incomplete knowledge of the physical system and system abnormal operating conditions. Robust controllers minimise the effect of external disturbances on system output in terms of a defined norm. This norm is defined such that it can put the various types of disturbances into a single framework [81, 82, 83].

Nonlinear control techniques have also been used to design PSSs [84, 85, 86, 87]. In reference [84] the concept of exact stochastic feedback linearisation is used to design a nonlinear excitation controller for a single machine-infinite bus power system. In reference [85] a combination of H_∞ control theory and an exact feedback linearisation technique has been used to set up an excitation control for a multimachine power system. Simulation studies showed that the proposed controllers can improve the dynamic performance of the power system and enhance its stability. Reference [86] presents a nonlinear variable structure and self-adjusting PSS. Reference [87] uses the differential geometric linearisation approach to design a PSS.

Adaptive PSSs, H_∞ control and nonlinear control techniques demonstrated that it is possible to achieve much better performance than the CPSS. However, most modern control techniques require extensive mathematical calculations, which implies the need for high speed processors and high implementation costs.

Recently artificial intelligence techniques have been applied to the design of the PSSs. One of these techniques is based on the use of ANNs [88, 89, 90]. Another possibility is the use of fuzzy logic based controllers. Fuzzy logic offers excellent prospects in control and, because of its simplicity and relatively low cost of implementation, is rapidly gaining popularity in the design and implementation of many practical controllers.

1.6 Fuzzy Logic Control

Unlike classical design approach which requires a deep understanding of the system, exact mathematical models and precise numeric values, fuzzy logic incorporates an alternative way of design. It provides a tool to control complex systems using a higher level of abstraction originated from accumulated knowledge and experience. Fuzzy logic control techniques have been found to be a better alternative to conventional control techniques where fairly accurate mathematical models are not accessible or difficult to obtain. By using fuzzy logic, the control system designer will be able to reduce development time and costs with good performance. A *fuzzy logic system* (FLS) is unique- in that it is able to simultaneously handle numerical data and linguistic knowledge. It is a nonlinear mapping of an input data vector into a scalar output. To date, a FLS is the only approximation method that is able to incorporate both types of knowledge in a unified mathematical manner [91]. By a proper design, a FLS can approximate any nonlinear function [92].

During the past several years, fuzzy logic control has emerged as one of the most active and fruitful areas for research in the applications of fuzzy set theory [93]. Fuzzy logic is a logical system which is much closer in spirit to human thinking and natural language than traditional logical systems. The *Fuzzy Logic Controller* (FLC) based on fuzzy logic provides a means of converting a linguistic control strategy based on expert knowledge into an automatic control strategy [94].

A basic feature of the FLC is that a process can be controlled without knowing the exact dynamics of the plant in terms of mathematical expressions. The operator can simply express the control strategy, learned through experience, by a set of rules. These rules describe the behaviour of the controller using linguistic terms. The controller then infers the proper control action from this rule base which emulates the role of the human operator or a bench-mark control action.

Such kind of knowledge exists in many industrial control processes. Obviously, the control rules are model-independent: no matter how (mathematically) complex the process is, an experienced operator can still give some control rules. Thus, FLCs are suitable for poorly-understood systems.

FLCs are easier to develop than conventional controllers. In the design of a conventional linear controller, the following steps are taken after selecting the sensors [95]:

1. **Modelling**: Build a mathematical model describing the process.

2. **Linearisation**: Linearise the model.

3. **Control design**: Solve system equations. Make a prototype design based on classical control criteria.

4. **Simulation**: Simulate the design. If not satisfied, go to step 1.

For a FLC, the steps are:

1. **Analysis**: Analyse the process. This analysis can be qualitative.

2. **Acquisition of rules**: Acquire control rules from experienced operators.

3. **Simulation**: Simulate the fuzzy controller. If not satisfied, go to step 1.

For processes that are difficult to model but have straightforward control rules, the FLCs are easy to design and implement. Since the fuzzy controllers are designed directly from the input-output properties of the process, the development time will be shorter for FLCs than for conventional controllers.

1.7 Application of Fuzzy Logic in the PSS Design

Fuzzy control has been applied to the design of power system stabilisers (PSSs) in a number of publications [96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109]. Using fuzzy logic control, researchers have been able to develop a new generation of effective, reliable and cost effective PSSs. In all these papers, after the design is completed, the parameters of the fuzzy power system stabiliser (FPSS) are kept constant. Recently, some attempts have been made to adjust the parameters of FPSSs with various techniques [110, 111]. Reference [110] has used fuzzy logic techniques to optimise and update the controller parameters on line. Reference [111] has used genetic algorithm to optimise the parameters of a FPSS.

The performance of the FPSS depends on the operating conditions of the power system, although it is less sensitive than conventional linear power system stabilisers (CPSS) [36].

1.8 Artificial Neural Networks

Research on artificial neural networks (ANNs) has advanced rapidly in recent years. Because of the various advantages they offer over the conventional computing systems, ANNs have attracted considerable attention as a candidate for novel computational systems. Use of ANNs in control systems has progressed at a steady pace during the past years. A good review on the applications of ANNs in control systems is given in references [112, 113, 114].

The advantages of ANNs are as follows:

1) They are adaptive: they take data and learn from it. Thus, they infer solutions from the data presented to them, often capturing quite subtle relationships. This ability differs radically from standard software techniques because it does not depend on the programmer's prior knowledge of rules. ANNs can reduce development time by learning underlying relationships even if they are difficult to find and describe. They can also solve problems that lack existing solutions.

2) ANNs can generalise: they can correctly process data that only broadly resembles the data they were trained on originally. Similarly, they can handle imperfect or incomplete data, providing a measure of fault tolerance. Generalisation is particularly useful in practical applications because real world data is noisy.

3) The ANNs are nonlinear: they can capture complex interactions among the input variables in a system.

4) ANNs are highly parallel: their numerous identical, independent operations can be executed simultaneously.

However, ANNs have some disadvantages as follows:

1) They can be difficult to account for their results. ANNs like human experts express opinions that can not be easily explained.

2) Choosing the optimum architecture for ANNs is not straightforward.

3) ANNs can consume large amounts of computer time during training.

1.8.1 Application of artificial neural networks in the PSS design

ANN applications in power engineering extend in many areas including security and contingency analysis, machine modelling and identification, fault diagnosis, harmonic source monitoring and identification, alarm processing, load forecasting, state estimation, economic load dispatch, etc. [115, 116, 117, 118, 119].

Compared with the applications in other areas of power systems, ANN application in power system stability control is still a new area. References [88, 90, 120, 121] had been published in this area, when this work (the thesis) started. In reference [88] an ANN was used to tune the parameters of a conventional PSS, in reference [90] an ANN was trained to simulate the function of an adaptive power system stabiliser, whilst in reference [120] an ANN was applied to tune a FPSS. The performances of the ANN PSSs introduced in references [88] and [90] are dependent on the CPSS or APSS. A multi-input ANN PSS was investigated in reference [121]; the generator speed deviation and the electrical power deviation as the plant's outputs together with their delayed values and the delayed stabilising signal in the excitation system were used as the inputs.

Research is going on in the area of applying ANNs to the power system stability control. Parallel to this thesis, references [122, 123, 124, 125, 126, 127, 128, 129, 130] have been published recently. In reference [122], similar to reference [88], an ANN is used to tune the parameters of a conventional PSS. The proposed

controller in reference [123] is first trained off-line using a pole placement based state feedback gain technique at different operating points. The trained ANN parameters are updated and tuned on-line using the speed deviation as the reinforcement signal. Reference [124] uses two ANNs called inverse dynamic neural network (IDNN) and error reduction network (ERN), respectively. The IDNN represents the inverse dynamic of the plant and requires training. The control error due to interactions between generators is predicted and compensated through the ERN. Reference [125] combines the advantages of the ANNs and fuzzy logic control to construct a PSS with learning ability. Reference [126] presents a coordinated excitation/governor ANN based controller. The proposed ANN based controller replaces the global action of the voltage regulator, PSS, and speed governor controls. Application of an ANN to integrate the AVR and the CPSS into a single controller is also described in reference [127]. An enhanced adaptive ANN control scheme, based on the adaptive linear element (Adaline), is proposed and tested by applying it to a multimachine power system in reference [128]. Reference [129] investigates the use of modular neural networks for PSS modelling. Modular neural networks learn different aspects of a problem by partitioning the data space into several different regions. A self-learning algorithm based on the learning capabilities of ANNs is proposed as a PSS in reference [130]. This algorithm aims to learn the inverse dynamics of a controlled system.

1.9 Scope of the Thesis

1.9.1 Motivation for the thesis

This thesis involves the use of fuzzy logic and ANNs to design effective PSSs. The objective is to incorporate the advantages of intelligent control to enhance the damping of dynamic and transient responses of an industrial cogenerator system.

Recent advances in intelligent control (specially control based on fuzzy logic and ANNs) have motivated power engineers to make use of different sorts of intelligent control to enhance power system operation. Although, there are reports of using fuzzy logic and ANNs in power system stability applications, lack of a complete survey and comparison of different schemes of PSS design using fuzzy logic and ANNs is observed. Historically, fuzzy logic PSSs have been employed with a static rule-based framework. ANNs have been mostly used to imitate other PSSs, e.g., CPSS or APSS.

There is a need for new concepts in fuzzy-logic and neural-network based control to be examined in the field of PSS design. Especially, different schemes to construct adaptive PSSs which make use of adaptive fuzzy logic controllers and ANNs should be investigated.

The following questions were raised which motivated the conductance of this research:

- If not enough information is available about the plant under study, how can fuzzy logic or ANNs be employed to design an effective PSS?
- If some linguistic rules are available from the experts, how can these rules be used to implement a reliable PSS?
- How can an optimum PSS be designed by combining fuzzy logic and ANNs to operate in a wide range of operation conditions?
- How can a FPSS be tuned using another fuzzy logic system?
- How can stable adaptive PSSs be designed using fuzzy logic or ANNs?
- After using various schemes to design PSSs based on fuzzy logic and ANNs, which of them has the best performance?

1.9.2 Original contributions of the thesis

The original contributions of this thesis are summarised as follows:

1. Two static FPSSs (fuzzy logic PSSs with fixed parameters) based on Mamdani rule-based fuzzy logic controller [131] and polar FPSS [102] have been designed and simulated.

2. Two real-time tuning schemes of the FPSS have been proposed. In the first scheme an ANN has been employed to tune the FPSS. The resultant PSS is called a neuro-fuzzy PSS (NFPSS). In the second scheme a fuzzy logic system (FLS) is used to tune the FPSS. The resultant PSS is called a tuned fuzzy logic PSS (TFPSS).

3. An adaptive PSS has been proposed and designed using a hierarchical structure of ANNs. It is called an adaptive neural network PSS (ANNPSS).

4. An indirect adaptive fuzzy logic PSS (IAFPSS) has been proposed and designed. The design is based on the Lyapunov's synthesis method [132]. Therefore, the stability of the system is guaranteed with this PSS. In the design procedure, it is assumed that not enough mathematical information is available about the nonlinear plant under study.

5. A direct adaptive fuzzy logic PSS (DAFPSS) has been designed using the Lyapunov's synthesis method with similar specifications to the IAFPSS. The DAFPSS needs less computations compared to the IAFPSS.

6. A performance index is defined and the performance of the proposed PSSs have been compared and discussed through nonlinear simulations.

1.9.3 Organisation of the thesis

This thesis consists of ten chapters. The remaining nine chapters are organised as follows:

A background of the theory of fuzzy logic control is given in Chapter 2. History of fuzzy logic, fuzzy logic systems, and fuzzy basis functions are discussed in this chapter.

Chapter 3 is an overview of power system modelling. Power system model for dynamic (steady state) stability and power system model for transient stability are discussed in this chapter.

A method for the design of a conventional PSS in the frequency domain is explained in Chapter 4.

Fuzzy logic PSSs with fixed parameters are discussed briefly in Chapter 5. Two methods are discussed, namely, a FPSS based on the Mamdani fuzzy controller and a polar FPSS following the proposal made by Hiyama.

The application of fuzzy logic and ANNs in the design of adaptive PSSs are discussed in the following chapters. In Chapter 6 two schemes for on-line tuning of the FPSS are proposed. The chapter covers the explanation of these two schemes, namely, on-line tuning using ANNs and on-line tuning using fuzzy logic systems.

Chapter 7 is dedicated to the design of an adaptive PSS by employing a hierarchical ANN. It explains how two artificial neural networks can be used to identify a power system plant and to stabilise the system adaptively such that it follows a desired response.

Adaptive fuzzy logic PSSs are proposed in Chapter 8. These PSSs are designed on the basis of Lyapunov's synthesis method. The stability of the system is

guaranteed by using these AFPSSs. Two schemes are proposed: a direct scheme and an indirect scheme. It is assumed that an exact mathematical model of the plant is not available in designing these AFPSSs.

The performances of the proposed PSSs are investigated and compared in Chapter 9. The system dynamic performance is inspected by applying a disturbance in the reference AVR voltage and a step change in the mechanical input power. The system transient performance is examined by applying a three-phase to ground fault at the sending end of the transmission line. Three different operating conditions and system configurations are considered for various tests.

The final concluding remarks are given in Chapter 10. Summary of results are discussed and suggestions for further research are given.

Chapter 2

Fuzzy Logic Control Theory and Background

"Most of human reasoning is approximate rather than exact. In a way that is not well understood at present, humans have a remarkable ability to make rational decisions in an environment of uncertainty and imprecision. We can understand distorted speech, decipher sloppy handwriting, park in a tight spot, understand poetry and summarise complex stories. In so doing, we perform no computations in the conventional sense of the term. We do manipulate information, which is what computation does, but the objects of our reasoning are generally not numbers but fuzzy patterns without sharply defined boundaries."

Lotfi A. Zadeh- The pioneer of Fuzzy Logic.

2.1 Introduction

One of the successful applications of fuzzy sets and systems theory to practical problems is fuzzy logic control. The present interest in fuzzy logic theory is largely due to the successful applications of fuzzy logic controllers (FLCs) to a variety of consumer products and industrial systems [133]. FLCs are very useful when an exact

mathematical model of the plant is not available, however, experienced human operators are available for providing qualitative rules to control the system.

During the past several years, fuzzy logic control has emerged as one of the most active and fruitful areas for research in the application of fuzzy set theory [93]. The pioneering research of Mamdani and his colleagues on fuzzy logic control [134, 135, 136, 137, 138] was motivated by Zadeh's seminal papers on the linguistic approach and system analysis based on the theory of fuzzy sets [139, 140, 141, 142]. Applications of fuzzy logic control in water quality control [143], automatic train operation systems [144], automatic container crane operation systems [145], nuclear reactor control [146], fuzzy logic controller hardware systems [147], fuzzy memory devices [148, 149, 150, 151], and fuzzy computers [152, 153] pointed a way for an effective utilisation of fuzzy logic control in the context of complex ill-defined processes that can be controlled by a skilled human operator without the knowledge of their underlying dynamics.

Fuzzy logic, which is the logic on which fuzzy logic control is based, is much closer in spirit to human thinking and natural language than the traditional logic systems. Basically, it provides an effective mean of capturing the approximate, inexact nature of our knowledge about the real world. Viewed in this perspective, the essential part of the fuzzy logic controller (FLC) is a set of linguistic control rules related by dual concepts of fuzzy implication and the compositional rule of inference. In essence, the FLC provides an algorithm which can convert the linguistic control strategy based on expert knowledge into an automatic control strategy. The methodology of the FLC appears very useful when the processes are too complex for analysis by conventional quantitative techniques [94].

2.2 History of Fuzzy Logic

The precision of mathematics owes its success in a large measure to the efforts of Aristotle and the philosophers who preceded him. In their efforts to devise a concise theory of logic the so-called "Laws of Thought" were suggested [154]. One of these, the "Law of the Excluded Middle", states that every proposition must either be *true* or *false*.

It was Plato who laid the foundation for what would later become fuzzy logic, indicating that there was a third region (other than true and false). Some other philosophers echoed his sentiments. However, it was Lukasiewicz, Polish logician, who first proposed a systematic alternative to the bivalent logic of Aristotle [155].

In the early 1930s, Lukasiewicz described a three-valued logic, along with the mathematics to accompany it. With the third value, he extended the range of truth values from {0, 0.5, 1} to all rational numbers in the interval [0, 1], and finally to all numbers in [0, 1]. Eventually, he proposed an entire notation and axiomatic system from which he hoped to derive modern mathematics.

Later, in 1937 Black, a quantum philosopher, published a paper called "Vagueness: an exercise in logic analysis" [156]. In this paper he expressed the need to bridge the gap between a mathematical model and experience. He defined the first simple fuzzy set with a curve that is now called a *membership function.*

The same need was expressed by Zadeh [157], when he was dealing with biological systems, three years before he actually proposed the new paradigm of mathematics based upon the concept of a fuzzy set:

> *... There are some who feel this gap reflects the fundamental inadequacy of the conventional mathematics- the mathematics of precisely-defined points,*

functions, sets, probability measures, etc.- for coping with the analysis of biological systems, and that to deal effectively with such systems, which are generally orders of magnitude more complex than man-made systems, we need a radically different kind of mathematics, the mathematics of fuzzy or cloudy quantities which are not describable in terms of probability distributions. ...

The notion of a multi-valued logic, called *fuzzy logic*, was introduced by Zadeh in 1965 [93]. The basis of the idea started early in 1950s, when he wrote an article about an electronic director of admission based on IF-THEN rules. The system was a primeval decision-maker expert system. This article declared the importance of multi-valued logic with values between true and false [158]. Zadeh believed that fuzzy logic would find home in psychology, philosophy, and human science. He suggested it would play an important role in control.

At first, fuzzy logic was not welcomed by many scholars. It faced a lot of resistance. One of the reasons behind this resistance was its name. Many people did not realise that fuzzy logic is not a logic that is fuzzy but *a logic that describes fuzziness*. The opposition against fuzzy logic increased in the early 1970s. Rudolph E. Kalman, the inventor of the Kalman filter, and William Kahan, a mathematician and colleague of Zadeh, were amongst those who opposed fuzzy logic [158].

In spite of the resistance against fuzzy logic by a number of scientists, fuzzy logic has found its place in well formulated theory and applications. In 1973 Zadeh published one of his most influential papers, which is a key paper outlining a new approach to the analysis of complex systems [142]. This paper laid the framework for fuzzy logic control. In fact it brought fuzzy logic out of the web of theoretical esoteric to the daylight. It showed how engineers could use fuzzy logic.

In the same year Mamdani and Assilian of Queen Mary College, in the

University of London, were looking for a device that could learn by itself to regulate the pressure of a steam engine. They succeeded in implementing the fuzzy if-then rules to control the steam engine. The results were better than using numerical methods and modelling. This application laid the ground for today's fuzzy logic control systems [131].

Fuzzy logic theory is actually an alternative view to the concept of *uncertainty* [159]. The traditional view of this concept insists that uncertainty is undesirable in science and should be avoided by all possible means. The alternative view is tolerant of uncertainty and insists that science cannot avoid it. According to the traditional view, science should strive for certainty in all its manifestations (precision, specifity, sharpness, consistency, etc.); hence, uncertainty (imprecision, nonspecifity, vagueness, inconsistency, etc.) is regarded as unscientific. According to the alternative (or modern) view, uncertainty is considered essential to science; it is not only an unavoidable plague, but it has a great utility. Klir [159] believes that the transition from the traditional view to the modern view of uncertainty has characteristics typical of processes, usually referred to as *scientific paradigm shifts* [160]. The paradigm shift initiated by the concept of a fuzzy set and the idea of mathematics based upon fuzzy sets has similar characteristics to other paradigm shifts recognised in the history of science.

The paradigm shift is still ongoing, and it will likely take much longer than usual to complete it. This is not surprising, since the new paradigm does not affect any particular field of science, but the very foundations of science. In fact it challenges the most sacred element of the foundations- the Aristotelian two-valued logic, which for millennia has been taken for granted [159].

2.3 Fuzzy Logic Systems

Block diagram of a *Fuzzy Logic System* (FLS) which is widely used in fuzzy logic controllers and signal processing applications is shown in Fig. 2.1 [91]. A FLS

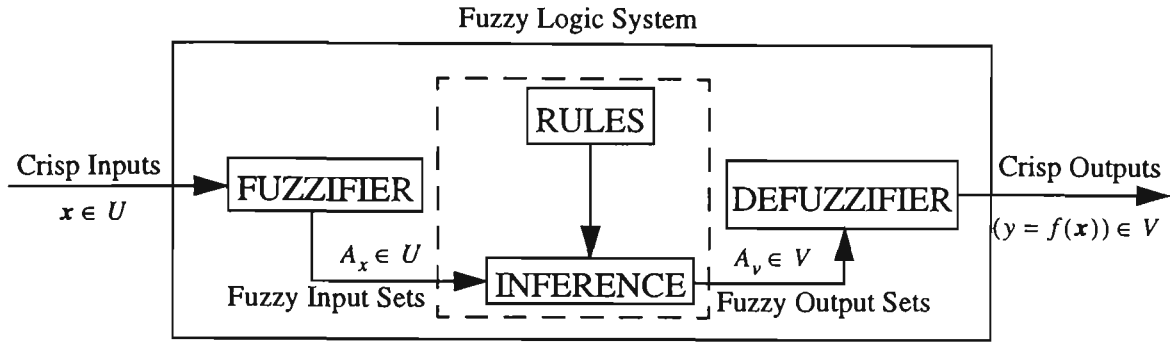


Fig. 2.1 Block diagram of a fuzzy logic system (FLS).

maps crisp inputs into crisp outputs. It contains four components: fuzzifier, rules base, inference engine, and defuzzifier. Once the rules have been established, a FLS can be viewed as a non-linear mapping from crisp inputs to crisp outputs, and this mapping can be expressed mathematically as $y = f(x)$.

The *fuzzifier* maps crisp numbers into fuzzy sets. It is needed in order to activate rules which are in terms of linguistic variables, which have fuzzy sets associated with them.

Rules may be provided by experts or can be extracted from numerical data. In either case, rules are expressed as a collection of IF-THEN statements, e.g., "IF speed-deviation is negative big and accelerating power is positive big, THEN controlling input should be around zero". This sample rule reveals that we will need an understanding of: 1) linguistic variables versus numerical values of a variable (e.g., *negative big* versus -5 rad/sec); 2) quantifying linguistic variables (e.g., speed-deviation may have a finite number of linguistic terms associated with it, ranging from *negative big* to *positive big*), which is done using *fuzzy membership*

functions; 3) logical connections for linguistic variables (e.g., "and", "or", etc.); 4) implications, i.e., "IF A THEN B". Additionally, we will need to understand how to combine two or more rules.

The *inference engine* of the FLS maps fuzzy sets into fuzzy sets. It handles the way in which rules are combined. Just as we humans use many different types of inferential procedures to help us understand things or to make decisions, there are many different fuzzy logic inferential procedures. Only a very small number of them are actually being used in engineering applications of fuzzy logic.

In many applications, crisp numbers must be obtained at the output of a FLS. The *defuzzifier* maps output sets into crisp numbers. In control applications, for example, such a number corresponds to a control action.

In what follows, the four elements of the FLS are explained in more detail, so that a mathematical formula can be derived that relates the output of the FLS to its inputs.

2.3.1 Fuzzification

The *fuzzifier* maps a crisp point $x_0 = [x_{01}, \ldots, x_{0p}]^T$ into a fuzzy set A_x in U. The most widely used fuzzifier in engineering applications is the *singleton fuzzifier* which is nothing more than a fuzzy singleton. A_x is a fuzzy singleton with support x_0 if $\mu_{A_x}(x) = 1$ for $x = x_0$ and $\mu_{A_x}(x) = 0$ for all other $x \in U$ when $x \neq x_0$. Basically, a fuzzy singleton is a precise value and hence no fuzziness is introduced by fuzzification in this case. This strategy has been widely used in fuzzy logic control applications since it is natural and easy to implement.

In the case that a singleton fuzzifier is used, the supremum operation in the

sup-star composition of equation (2.10) disappears, i.e.,

$$\mu_{B'}(y) = sup_{x \in A_x}[\mu_{A_x}(x) \bigstar \mu_{A \to B}(x, y)] = \mu_{A \to B}(x_0, y) \tag{2.1}$$

So, in this case, if a minimum implication is used

$$\mu_{B'}(y) = min(\mu_{F_1'}(x_{01}), \ldots, \mu_{F_p'}(x_{0p}), \mu_{G'}(y)) \tag{2.2}$$

and if a product implication is used

$$\mu_{B'}(y) = \mu_{F_1'}(x_{01}) \ldots \mu_{F_p'}(x_{0p}) \mu_{G'}(y) \tag{2.3}$$

Singleton fuzzification may not always be adequate, especially when data is corrupted by measurement noise. Non-Singleton fuzzification provides a means for handling such uncertainties totally within the framework of the FLS. In reference [162] a *non-singleton fuzzifier* is one for which $\mu_{A_x}(x) = 1$ for $x = \bar{x}_0$, where $\bar{x}_0$ is the mean value of the noisy input data, and $\mu_{A_x}(x)$ decreases from unity as x moves away from $\bar{x}_0$. This non-singleton fuzzifier has a triangular fuzzy membership function, whose vertex corresponds to the mean value of the input data. The base of the triangle is twice the standard deviation of the noisy input data.

2.3.2 Rules

A *fuzzy rule base* consists of a set of linguistic rules in the form of "IF a set of conditions are satisfied, THEN a set of consequences are inferred". Since the

antecedents and the consequents of these IF-THEN rules are associated with fuzzy concepts (linguistic terms), they are often called *fuzzy conditional statements*. In the fuzzy logic control terminology, a *fuzzy logic control rule* is a fuzzy conditional statement in which the antecedent is a condition in its application domain and the consequent is a control action for the system under control. Basically, fuzzy logic control rules provide a convenient way for expressing control policy and domain knowledge. The fuzzy rule base is a collection of IF-THEN rules, which can be expressed as:

$$R^{(l)}: \text{IF } u_1 \text{ is } F_1^l \text{ and } u_2 \text{ is } F_2^l \text{ and} \ldots u_p \text{ is } F_p^l, \text{ THEN } v \text{ is } G^l \tag{2.4}$$

where $l = 1, 2, \ldots, M$, F_i^l and G^l are fuzzy sets in $U_i \subset R$ and $V \subset R$, respectively (R denotes the set of real numbers), $\boldsymbol{u} = \left[u_1\ u_2\ \ldots u_p\right]^T \in U_1 \times U_2 \times \ldots \times U_p$ is the input vector, and $v \in V$ is the output. $\boldsymbol{u}$ and v are linguistic variables. Their numerical values are $x \in U$ and $y \in V$, respectively. F_i^l and G^l are quantified through fuzzy membership functions defined for each of them.

In the case of a two-element input vector for a PSS, an example of a fuzzy IF-THEN rule is:

IF the ***speed deviation*** is ***negative small*** and the ***accelerating power*** is ***negative medium***, THEN the PSS ***controlling signal*** should be ***negative big***.

here u_1 = *speed deviation*, F_1^l = *negative small*, u_2 = *accelerating power*, F_2^l = *negative medium*, v = *controlling signal*, and G^l = *negative big*.

2.3.3 Fuzzy Inference Engine

In the fuzzy inference engine fuzzy logic principles are used to combine fuzzy IF-THEN rules into a mapping from fuzzy input sets in $U = U_1 \times U_2 \times \ldots \times U_p$ to fuzzy output sets in V. Each rule is interpreted as a *fuzzy implication* $F_1^l \times F_2^l \times \ldots \times F_p^l \rightarrow G^l$, which is a fuzzy set in $U \times V$ with a membership degree

$$\mu_{F_1^l \times \ldots \times F_p^l \rightarrow G^l}(x_1, \ldots, x_p, y) = \mu_{F_1^l}(x_1) \star \ldots \star \mu_{F_p^l}(x_p) \star \mu_{G^l}(y) \qquad (2.5)$$

Let $F_1^l \times F_2^l \times \ldots \times F_p^l \triangleq A$ and $G^l \triangleq B$, then

$$\mu_{F_1^l \times \ldots \times F_p^l \rightarrow G^l}(x_1, \ldots, x_p, y) = \mu_{A \rightarrow B}(x, y) \qquad (2.6)$$

The operator "★" stands for a *triangular norm* or *T-norm* [159]. The most commonly used operations for a T-norm are "minimum" and "product". Thus the membership degree corresponding to *minimum implication* is

$$\mu_{A \rightarrow B}(x, y) = min(\mu_{F_1^l}(x_1), \ldots, \mu_{F_p^l}(x_p), \mu_{G^l}(y)) \qquad (2.7)$$

and the membership degree corresponding to *product implication* is

$$\mu_{A \rightarrow B}(x, y) = \mu_{F_1^l}(x_1) \ldots \mu_{F_p^l}(x_p) \mu_{G^l}(y) \qquad (2.8)$$

Suppose that x_0 is the crisp value of the input vector to the FLS at a specific

instant of time. After fuzzification it will convert to a fuzzy set A_x whose membership function is

$$\mu_{A_x}(x_0) = \mu_{A_{x1}}(x_{01}) \star \ldots \star \mu_{A_{xp}}(x_{0p}) \tag{2.9}$$

where $A_{xk} \subset U_k (k = 1, \ldots, p)$ are the fuzzy sets describing the inputs.

Each rule $R^{(l)}$ determines a fuzzy set $B^l = A_x \circ R^{(l)}$, in which operator o stands for the *composition* operator. The membership degree of this fuzzy set is determined by

$$\mu_{B^l}(y) = sup_{x \in A_x}[\mu_{A_x}(x) o \mu_{A \to B}(x, y)] \tag{2.10}$$

where y is the numerical value of the output of the FLS and *sup* is the supremum operator which will be equal to the *max* operator in the case of discretised inputs and outputs. Equation (2.10) is called the *sup-star composition.* It is the input-output relationship in Fig. 2.1 between the fuzzy set that excites a one-rule inference engine and the fuzzy set at the output of that engine.

The final fuzzy set $A_v = A_x o[R^{(1)}, R^{(2)}, \ldots, R^{(M)}]$, which is determined by all the rules in the rule base, is obtained by combining B^l and its associated membership function for all $l = 1, 2, \ldots, M$. Normally, in engineering applications, the rules are connected using the fuzzy union operation, i.e., $A_v = B^1 \oplus B^2 \oplus \ldots \oplus B^M$ with the membership degree

$$\mu_V(y) = max(\mu_{B^1}(y), \mu_{B^2}(y), \ldots, \mu_{B^M}(y)) \tag{2.11}$$

The symbol "⊕" stands for the union operator.

It is also possible to combine rules additively. Kosko calls such a FLS (with an appropriate fuzzifier and defuzzifier) an *additive FLS* [161]. The membership degree of the output of the fuzzy inference engine in such an additive FLS will be calculated as

$$\mu_V(y) = \frac{w_1\mu_{B^1}(y) + w_2\mu_{B^2}(y) + \ldots + w_M\mu_{B^M}(y)}{w_1 + w_2 + \ldots + w_M} \tag{2.12}$$

where $w_1, w_2, \ldots, w_M$ of the combiner can be thought of as providing degrees of belief to each rule. In the case that some rules are more reliable than others, such rules would be assigned larger weights than less reliable rules. If such information is not known in advance, it is possible to either set all the weights equal to unity or use a training procedure to learn optimal values for the weights.

2.3.4 Defuzzification

The *defuzzifier* produces a crisp output for the FLS from the fuzzy set that is the output of the inference engine in Fig. 2.1. Many defuzzifiers have been proposed in the literature [163]. In engineering applications, one criterion for the choice of a defuzzifier is *computational simplicity*. This criterion has led to the following candidates for defuzzifiers:

2.3.4.1 Maximum Defuzzifier

This defuzzifier examines the fuzzy set V and chooses as its output the value of y for which $\mu_V(y)$ is a maximum. It can lead to peculiar results or can hung up. The former occurs for the situation depicted in Fig. 2.2(a), in which the maximum defuzzifier would choose $y = 1$ as the output of the FLS; this value totally ignores the fact that $\mu_V(y)$ is distributed from $y = 0.2$ to $y = 1$. The latter occurs for the situation depicted in Fig. 2.2(b), in which the maximum of $\mu_V(y)$ occurs for a range of y values rather than at a unique point.

2.3.4.2 Mean of Maxima Defuzzifier

This defuzzifier examines the fuzzy set A_v and first determines the values of y for which $\mu_V(y)$ is a maximum. It then computes the mean of these values as its output. Unfortunately, it can also lead to some peculiar results. If the maximum value of $\mu_V(y)$ only occurs at a single point, then the mean of maxima defuzzifier reduces to the maximum defuzzifier, and the discussion about Fig. 2.2(a) applies to it. Another situation that makes the mean of maxima defuzzifier fail to have a good result is depicted in Fig. 2.3. The output fuzzy set consists of two separated triangles

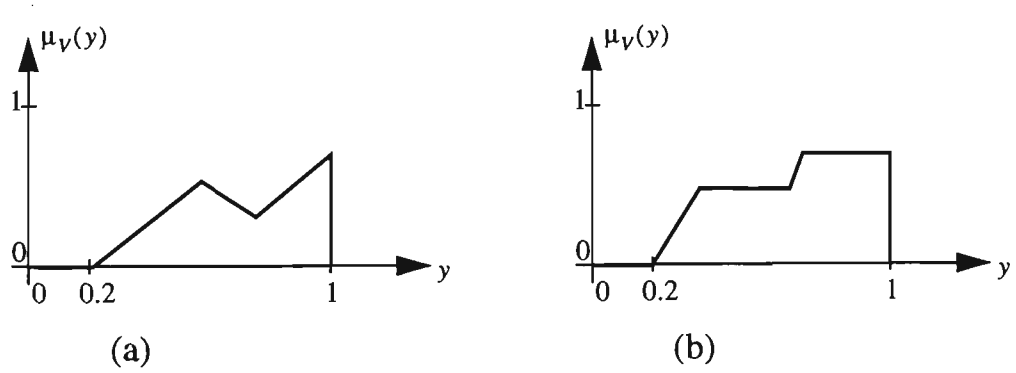


Fig. 2.2 The overall fuzzy set obtained at the output of fuzzy inference engine

that both have the same peak amplitudes. The mean of maxima defuzzifier assign a crisp value $y_0 = \frac{y_1 + y_2}{2}$ to the output of FLS, at which point the membership function $\mu_V(y)$ has zero value. This makes no engineering sense!

2.3.4.3 Centroid Defuzzifier

This defuzzifier determines the centre of gravity (centroid), $\bar{y}$, of A_v and uses this value as the output of the FLS. From calculus, it is known that

$$\bar{y} = \frac{\int_S y\mu_V(y)dy}{\int_S \mu_V(y)dy} \tag{2.13}$$

where S denotes the support of $\mu_V(y)$. Frequently, S is discretized, so that $\bar{y}$ can be approximated by the following formula which uses summations instead of integrations:

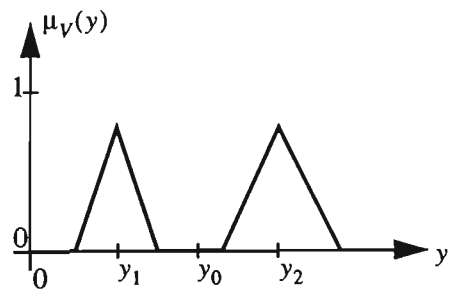


Fig. 2.3 Example for which mean of maximum defuzzification makes no sense

$$\bar{y} = \frac{\sum_{i=1}^{I} y_i \mu_V(y_i)}{\sum_{i=1}^{I} \mu_V(y_i)} \tag{2.14}$$

The centroid defuzzifier is unique; however, it is usually difficult to compute. Kosko et. al. has interpreted $\bar{y}$ as a conditional expectation [164].

Pacini and Kosko [165] prove that for product inference and additive combining of rules, $\bar{y}$ can be computed using centroid information about the individual M rules. While this result does not extend to other t-norms and t-conorms, it does provide some adhoc justification for what is probably the most widely used form of defuzzification, namely, the *height defuzzification.*

2.3.4.4 Height Defuzzifier:

Let $\bar{y}^l$ denote the centre of gravity of the fuzzy set B^l (which is associated with the activation of rule $R^{(l)}$). The height defuzzifier first evaluates $\mu_{B^l}(y)$ at $\bar{y}^l$ and then computes the output of the FLS as

$$y_h = \frac{\sum_{l=1}^{M} \bar{y}^l \mu_{B^l}(\bar{y}^l)}{\sum_{l=1}^{M} \mu_{B^l}(\bar{y}^l)} \tag{2.15}$$

It is very easy to use equation (2.15) because the centres of gravity of commonly used membership functions are known ahead of time. Centre of gravity of

three commonly used membership functions are shown in Table 2.1. The centre of gravity of B^l in equation (2.15) is given by Table 2.1, regardless of whether minimum or product inference is used. Equations (2.14) and (2.15) look alike, but they are different.

Although (2.15) is easy to use, it suffers from a deficiency that is not obvious to the newcomer. Whereas y_h makes use of the entire shape of each antecedent's membership function, because this information is embodied in $\mu_{B^l}(\bar{y}^l)$, it does not make use of the entire shape of the consequent membership function. It only uses the centre of gravity of each consequent membership function. Regardless of whether or not the consequent membership function is very narrow, which indicates a very strong belief in that rule, or is very broad, which indicates much less belief in that rule, the height defuzzifier gives the same result. This has led to the *modified height defuzzifier*.

2.3.4.5 Modified Height Defuzzifier

As in the height defuzzification, $\bar{y}^l$ stands for the centre of gravity of the fuzzy set B^l. The modified height defuzzifier [163] first evaluates $\mu_{B^l}(\bar{y}^l)$ and then computes the output of the FLS as

Table 2.1 Centre of gravity of commonly used membership functions

	Type of membership function		
	Symmetric triangular	Symmetric trapezoidal	Gaussian
Centre of gravity	Apex of the triangle	The midpoint of its support	Centre value of the Gaussian function

$$y_{mh} = \frac{\sum_{l=1}^{M} (\bar{y}^l \mu_{B^l}(\bar{y}^l))/(\sigma^l)^2}{\sum_{l=1}^{M} (\mu_{B^l}(\bar{y}^l))/(\sigma^l)^2} \tag{2.16}$$

where σ^l is a measure of the spread of the consequent for rule $R^{(l)}$. For triangular and trapezoidal membership functions, σ^l could be the support of the triangle or trapezoid, whereas, for Gaussian membership functions, σ^l could be its standard deviation. The modified height defuzzifier is also easy to use, although the σ^l parameters must be specified as well as $\bar{y}^l$ and $\mu_{B^l}(\bar{y}^l)$.

2.3.5 Possibilities in Choosing FLS Configuration

From the detailed discussions about the four elements which comprise the FLS, it is seen that there are many possibilities to choose from. The designer must decide on the type of fuzzification (singleton or non-singleton), functional forms for membership functions (triangular, trapezoidal, Gaussian, piecewise linear), parameters of membership functions (fixed in advance, tuned during a training procedure), composition (max-min, max-product), inference (minimum, product), and defuzzifier (centroid, height, modified height). Just choosing from the parenthetical possibilities leads to $2^{15} = 32768$ different FLS's.

2.4 Fuzzy Basis Functions

It is possible to describe a complete FLS with a mathematical formula that maps a crisp input x into a crisp output $y = f(x)$. From Fig. 2.1, it is observed that such a formula can be obtained by following the signal x through the FLS. The crisp

input x is converted to a fuzzy set A_x, when it passes through the fuzzifier. It is converted to a fuzzy set A_y by passing through the inference block. Finally it is converted to $f(x)$ after passing through the defuzzifier. In order to write such a formula, specific choices should be made for fuzzifier, membership functions, composition, inference, and defuzzifier.

For *singleton* fuzzification, *max-product* composition, *product* inference, and *height* defuzzification, leaving the choice of membership function open, it is easy to show that [166]:

$$y = f(\boldsymbol{x}) = \frac{\sum_{l=1}^{M} \bar{y}^l \prod_{i=1}^{p} \mu_{F_i^l}(x_i)}{\sum_{l=1}^{M} \prod_{i=1}^{p} \mu_{F_i^l}(x_i)} \tag{2.17}$$

In order to obtain equation (2.17), $\mu_{B^l}(\bar{y}^l)$ from equation (2.3) is substituted into equation (2.15). Equation (2.3) is written as

$$\mu_{B^l}(\bar{y}^l) = \mu_{F_1^l}(x_{01}) \ldots \mu_{F_p^l}(x_{0p}) \mu_{G^l}(\bar{y}^l) = \left[\prod_{i=1}^{p} \mu_{F_i^l}(x_{0i}) \right] \mu_{G^l}(\bar{y}^l) \tag{2.18}$$

It is assumed that membership functions are normalised, so that $\mu_{G^l}(\bar{y}^l) = 1$. Furthermore, for notational simplicity, x_{0i} is relabelled to x_i, so that $f(\boldsymbol{x_0})$ is written as $f(\boldsymbol{x})$.

For other selections of the elements of the FLS, similar formulas will be

derived. When Gaussian membership functions are used,

$$\mu_{F_i^l}(x_i) = e^{-[(x_i - \bar{x}_i^l)/\sigma_i^l]^2} \tag{2.19}$$

where $i = 1, 2, ..., p$ and $l = 1, 2, ..., M$ (recall that p equals the dimension of $\boldsymbol{x}$, and M equals the number of rules).

The FLS in equation (2.17) can also be represented as

$$y = f(\boldsymbol{x}) = \sum_{l=1}^{M} \bar{y}^l \phi_l(\boldsymbol{x}) \tag{2.20}$$

where $\phi_l(\boldsymbol{x})$ are called *fuzzy basis functions* (FBFs) [166] and are given by

$$\phi_l(\boldsymbol{x}) = \frac{\prod_{i=1}^{p} \mu_{F_i^l}(x_i)}{\sum_{l=1}^{M} \prod_{i=1}^{p} \mu_{F_i^l}(x_i)} \tag{2.21}$$

where $l = 1, 2, ..., M$. Now the FLS can be referred to as a fuzzy basis function expansion. Doing this is very useful, because it places a FLS into the more global perspective of function approximation. However, the FBF in equation (2.21) is valid only for very specific choices made for fuziffier, membership functions, composition, inference and defuzzifier. Although by changing any of these elements equation (2.21) will no longer be valid, the interpretation of a FLS as a fuzzy basis

function expansion still remains valid. Formulas that are comparable to equation (2.21) can be derived for many other possibilities.

The relationships between FBFs and other basis functions have been extensively studied in reference [167]. They are more general than radial basis functions, generalised radial basis functions, and hyper-basis functions [168]. For very special choices of their parameters, they bare structural resemblance to generalised regression neural networks [169] and Gaussian sum approximations [170].

The denominator in equation (2.21), which is a result of the height defuzzifier, serves to normalise the numerators of the FBFs. Generally, FBFs are nonlinear functions of radial basis functions. Equally spaced FBFs are possible only if the mean values (centres) of the antecedent membership functions can be chosen by the designer. If these values are estimated by means of a training procedure, then unequally spaced FBFs will be created. In the case that numerator of (2.21) is radially symmetric, FBFs can be referred to as normalised radial basis functions [91].

In the design of *adaptive fuzzy logic power system stabilisers* (AFPSS) proposed in this thesis, the concept of FBFs will be used (refer to Chapter 8).

Chapter 3

Overview of Power System Modelling

3.1 Overview of the Chapter

In this chapter, the mathematical models used for the synchronous machine are explained. The models suitable for small-signal stability and transient stability problems are described. These models will be used in simulation studies.

3.2 Introduction

In this chapter the modelling aspects of the power system under study will be discussed. Although the models introduced in this chapter have been thoroughly discussed by power engineers in the past [2], an overview of the models used in simulation studies is necessary. This chapter is mostly based on the work done by Kundur [1].

PSS's designed in this thesis will be applied to industrial cogenerator systems. A cogenerator combines the generation of heat and electricity in a single unit in a way that is more efficient than producing heat and electricity separately in boiler plant and at the power station. In other words, cogeneration is the energy process

whereby waste heat, produced during the generation of electricity, is utilised for steam raising or heating [171].

Security of supply is of paramount importance in industrial environments. An on-site cogeneration scheme can enhance the security of both heat and electricity supplies. In particular, it is possible to design the electrical connections to ensure continuity of supply for the complete failure of the Grid.

A primary design criteria for parallel connection of a cogenerator to the power pool is its ability to maintain stability on the occurrence of a power system disturbance that can result in unstable operation. The ability of the cogenerator to maintain stability under all operating conditions must be considered at the design stage. Operating conditions to be considered will include system normal as well as small and large scale disturbances such as short-circuit type faults.

From the viewpoint of a cogenerator, the rest of the system is large enough to be considered as an infinite bus. Therefore, one-machine infinite-bus system will be modelled for two different problems: small-signal stability and transient stability.

3.3 Small-Signal Stability and Transient Stability

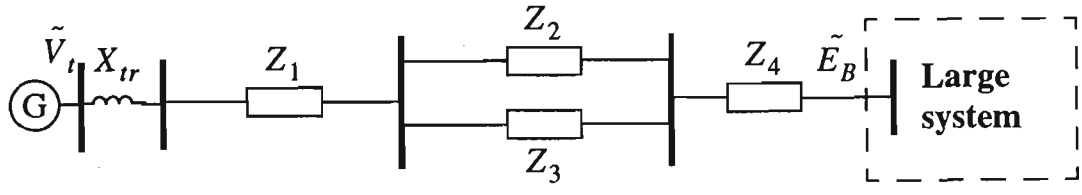
Small-signal stability is the ability of the power system to maintain synchronism when subjected to small disturbances [1]. In this context, a disturbance is considered to be small if by linearising the equations that describe the resulting response of the system a relatively accurate analysis can be performed. Instability that may result can be of two forms: (a) steady increase in generator rotor angle due to lack of synchronising torque [10], or (b) rotor oscillations of increasing amplitude due to lack of sufficient damping torque [10]. In today's practical power systems, the small-signal stability problem is usually due to insufficient damping of system

oscillations [1]. For the small-signal stability problem the procedure of deriving a linear model for the power system will be explained.

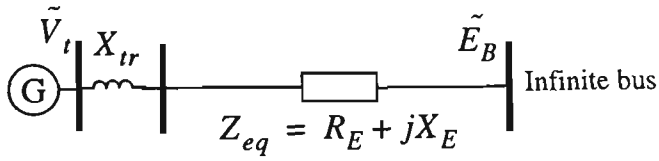
Transient stability is the ability of the power system to maintain synchronism when subjected to a severe disturbance such as a fault on transmission lines, loss of generation, or loss of a large load. The system response to such disturbances involves large excursions of generator rotor angles, power flows, bus voltages, and other system variables. Transient stability is influenced by the nonlinear characteristics of the power system. If the rotor angles of the machines in the system remain within a certain bound, the system maintains synchronism. If loss of synchronism occurs, it will be usually evident within 2 to 3 seconds of the initial disturbance.

3.4 Power System Model for Small-Signal Stability

In this section, the small-signal performance of a synchronous machine connected to a large system through transmission lines is reviewed. A general system configuration is shown in Fig. 3.1(a). Analysis of systems having such simple



(a) General configuration



(b) Equivalent system

Fig. 3.1 Single machine connected to a large system through transmission lines

configurations is very useful in understanding basic effects and concepts. For the purpose of analysis, the system of Fig. 3.1(a) may be reduced to the form of Fig. 3.1(b).

In the following sections, the small-signal stability of the system of Fig. 3.1(b) will be analysed. At first the synchronous machine will be represented by the classical model. Then, the model details will be increased to account for the effects of the dynamics of the field circuit and the excitation system. The block diagram representation and torque-angle relationships will be used to analyse the system-stability characteristics. The block diagram approach was first used by Heffron and Philips [172] and later by deMello and Concordia [10] to analyse the small-signal stability of synchronous machines.

3.4.1 Generator represented by the classical model

With the generator represented by the classical model and all resistances neglected, the system representation is as shown in Fig. 3.2. Here $\tilde{E}'$ is the voltage behind X_d', which is the direct-axis transient reactance of the generator. The magnitude of $\tilde{E}'$ is assumed to remain constant at the pre-disturbance value. Let δ be the angle by which $\tilde{E}'$ leads the infinite bus voltage $\tilde{E}_B$. A phasor diagram which

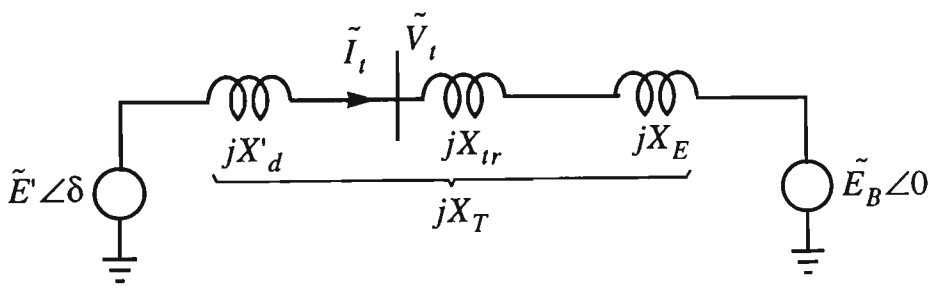


Fig. 3.2 The equivalent system when the generator is represented by the classical model

shows the relative positions of machine quantities is shown in Fig. 3.3. As the rotor oscillates during a disturbance, δ changes.

For this model, it can be shown [1] that

$$\frac{d}{dt}\begin{bmatrix}\Delta\omega_r \\ \Delta\delta\end{bmatrix} = \begin{bmatrix}-\frac{K_D}{2H} & -\frac{K_S}{2H} \\ \omega_0 & 0\end{bmatrix}\begin{bmatrix}\Delta\omega_r \\ \Delta\delta\end{bmatrix} + \begin{bmatrix}\frac{1}{2H} \\ 0\end{bmatrix}\Delta T_m \tag{3.1}$$

where $\Delta\omega_r$ is the per unit angular speed deviation of the rotor, H is the per unit inertia constant, T_m is the applied mechanical torque, K_D is the damping factor, δ is

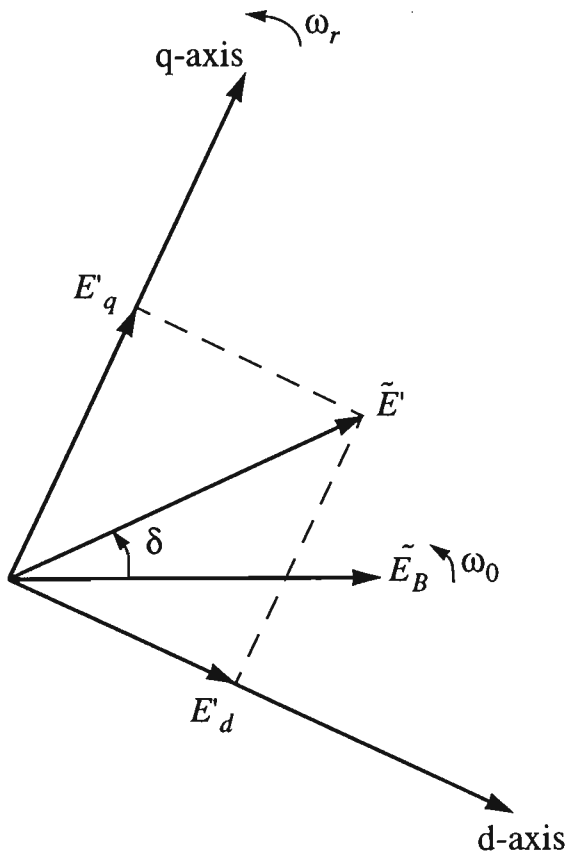


Fig. 3.3 Phasor diagram of machine quantities for the classical model

rotor angle in electrical radians, ω_0 is the base rotor electrical speed in radians per second, and K_S is the synchronising torque coefficient given by

$$K_S = \frac{E'E_B}{X_T}\cos\delta_0 \tag{3.2}$$

This is the state-space representation of the system in the form $\dot{x} = Ax + bu$. The elements of the state matrix A are seen to be dependent on the system parameters H, K_D, X_T, and the initial operating condition represented by the values of $\tilde{E}'$ and δ_0. Vector b is also dependent on H.

3.4.2 Effects of synchronous machine field circuit dynamics

In this subsection the system state-space model including the effect of field flux variations will be developed. The effect of excitation system will be considered in the next subsection.

As in the case of the classical generator model, the linearised equations of motion are

$$\frac{d}{dt}\Delta\omega_r = \frac{1}{2H}(\Delta T_m - \Delta T_e - K_D\Delta\omega_r) \tag{3.3}$$

$$\frac{d}{dt}\Delta\delta = \omega_0\Delta\omega_r \tag{3.4}$$

where ΔT_e is the electrical (air-gap) torque. In this case, the rotor angle δ is the

angle (in electrical radians) by which the q-axis leads the reference $\tilde{E}_B$. Phasor diagram which shows the relative positions of synchronous machine variables is shown in Fig. 3.4. The rotor angle is given by

$$\delta = \delta_t + \delta_i \tag{3.5}$$

where δ_t is the angle by which the terminal voltage phasor $\tilde{V}_t$ leads the reference $\tilde{E}_B$ and the steady state value of δ_i is given by

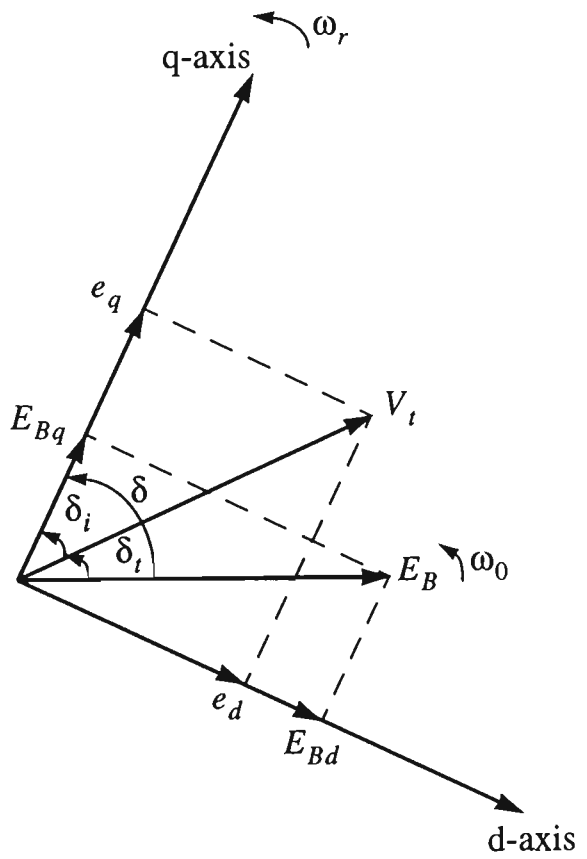


Fig. 3.4 Phasor diagram showing the relative positions of machine quantities

$$\delta_i = \text{atan}\left(\frac{X_q I_t \cos\varphi - R_a I_t \sin\varphi}{V_t + R_a I_t \cos\varphi + X_q I_t \sin\varphi}\right) \tag{3.6}$$

where V_t and I_t are terminal voltage and current, respectively. φ is the power factor angle, R_a is the armature resistance per phase and X_q is the quadrature-axis synchronous reactance.

The effect of field flux variations can be represented as

$$\frac{d}{dt}\Delta\psi_{fd} = \frac{\omega_0 R_{fd}}{L_{adu}}\Delta E_{fd} - \omega_0 R_{fd}\Delta i_{fd} \tag{3.7}$$

where ψ_{fd} is the rotor circuit (field) flux linkage, R_{fd} is the rotor circuit resistance, L_{adu} is the unsaturated direct-axis mutual inductance between stator and rotor windings, E_{fd} is the exciter output voltage, and i_{fd} is the field circuit current. Equations (3.3) to (3.7) describe the dynamics of the synchronous machine with $\Delta\omega_r$, $\Delta\delta$, and $\Delta\psi_{fd}$ as the state variables. However, the derivatives of these state variables appear in these equations as functions of ΔT_e and Δi_{fd}, which are neither state variables nor input variables. In order to develop the complete system equations in the state-space form, ΔT_e and Δi_{fd} should be expressed in terms of the state variables as determined by the machine flux linkage equations and network equations. It can be shown [1] that

$$\Delta i_{fd} = \frac{1}{L_{fd}}\left(1 + m_2 L'_{ads} - \frac{L'_{ads}}{L_{fd}}\right)\Delta\psi_{fd} + \frac{1}{L_{fd}} m_1 L'_{ads}\Delta\delta \tag{3.8}$$

where L_{fd} is the self-inductance of the field circuit,

$$L'_{ads} = \frac{L_{ads}L_{fd}}{L_{ads}+L_{fd}} \tag{3.9}$$

$$m_1 = \frac{E_B(X_{Tq}\sin\delta_0 - R_T\cos\delta_0)}{D} \tag{3.10}$$

$$m_2 = \frac{X_{Tq}}{D}\frac{L_{ads}}{(L_{ads}+L_{fd})} \tag{3.11}$$

$$\begin{aligned} R_T &= R_a + R_E \\ X_{Tq} &= X_{tr} + X_E + (L_{aqs} + L_l) \\ X_{Td} &= X_{tr} + X_E + (L'_{ads} + L_l) \\ D &= R_T^2 + X_{Tq}X_{Td} \end{aligned} \tag{3.12}$$

where X_{tr} is the reactance of the transformer, R_E is the Thevenin resistance of the network, X_E is the Thevenin reactance of the network, R_a is the armature resistance, L_l is the leakage inductance of the stator, L_{ads} is the saturated d-axis mutual inductance between stator and rotor windings, and L_{aqs} is the saturated q-axis mutual inductance between stator and rotor windings.

Also it can be shown that

$$\Delta T_e = K_1\Delta\delta + K_2\Delta\psi_{fd} \tag{3.13}$$

where

$$K_1 = n_1(\psi_{ad0} + L_{aqs} i_{d0}) - m_1(\psi_{aq0} + L'_{ads} i_{q0}) \tag{3.14}$$

$$K_2 = n_2(\psi_{ad0} + L_{aqs} i_{d0}) - m_2(\psi_{aq0} + L'_{ads} i_{q0}) + \frac{L'_{ads}}{L_{fd}} i_{q0} \tag{3.15}$$

where

$$n_1 = \frac{E_B(R_T \sin\delta_0 - X_{Td} \cos\delta_0)}{D} \tag{3.16}$$

$$n_2 = \frac{R_T}{D} \frac{L_{ads}}{(L_{ads} + L_{fd})} \tag{3.17}$$

$$i_{d0} = \frac{X_{Tq}\left[\psi_{fd0}\left(\frac{L_{ads}}{L_{ads} + L_{fd}}\right) - E_B \cos\delta_0\right] - R_T E_B \sin\delta_0}{D} \tag{3.18}$$

$$i_{q0} = \frac{R_T\left[\psi_{fd0}\left(\frac{L_{ads}}{L_{ads} + L_{fd}}\right) - E_B \cos\delta_0\right] - X_{Td} E_B \sin\delta_0}{D} \tag{3.19}$$

$$\psi_{ad0} = L'_{ads}\left(\frac{\psi_{fd0}}{L_{fd}} - i_{d0}\right) \tag{3.20}$$

$$\psi_{aq0} = -L_{aqs} i_{q0} \tag{3.21}$$

By substituting the expressions for Δi_{fd} and ΔT_e given by equations (3.8) and (3.13) into equations (3.3) and (3.7), the state-space representation of the system is obtained as follows:

$$\frac{d}{dt}\begin{bmatrix}\Delta\omega_r\\ \Delta\delta\\ \Delta\psi_{fd}\end{bmatrix} = \begin{bmatrix}a_{11} & a_{12} & a_{13}\\ a_{21} & 0 & 0\\ 0 & a_{32} & a_{33}\end{bmatrix}\begin{bmatrix}\Delta\omega_r\\ \Delta\delta\\ \Delta\psi_{fd}\end{bmatrix} + \begin{bmatrix}b_{11} & 0\\ 0 & 0\\ 0 & b_{32}\end{bmatrix}\begin{bmatrix}\Delta T_m\\ \Delta E_{fd}\end{bmatrix} \tag{3.22}$$

where

$$a_{11} = -\frac{K_D}{2H} \tag{3.23}$$

$$a_{12} = -\frac{K_1}{2H} \tag{3.24}$$

$$a_{13} = -\frac{K_2}{2H} \tag{3.25}$$

$$a_{21} = \omega_0 = 2\pi f_0 \tag{3.26}$$

$$a_{32} = -\frac{\omega_0 R_{fd}}{L_{fd}} m_1 L'_{ads} \tag{3.27}$$

$$a_{33} = -\frac{\omega_0 R_{fd}}{L_{fd}}\left(m_2 L'_{ads} + 1 - \frac{L'_{ads}}{L_{fd}}\right) \tag{3.28}$$

$$b_{11} = \frac{1}{2H} \tag{3.29}$$

$$b_{32} = \frac{\omega_0 R_{fd}}{L_{adu}} \tag{3.30}$$

ΔT_m and ΔE_{fd} depend on prime-mover and excitation controls. With constant mechanical input torque, $\Delta T_m = 0$.

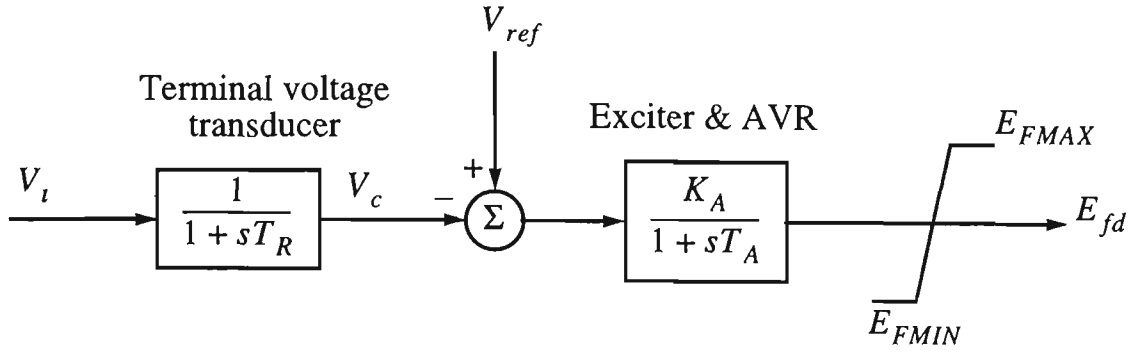


Fig. 3.5 Thyristor excitation system

3.4.3 Effects of excitation system

The input control signal to the excitation system is normally the generator terminal voltage $\tilde{V}_t$. Fig. 3.5 shows a simplified model of IEEE type ST1A thyristor (static) excitation system. The only nonlinearity associated with the model is due to the exciter output voltage limiter represented by E_{FMAX} and E_{FMIN}. For small-signal studies, these limits are ignored because the disturbances are so small that E_{fd} is always within the limits. It is seen from Fig. 3.5 that E_{fd} in Laplace domain can be given as

$$E_{fd} = \frac{K_A}{1+sT_A}(V_{ref} - V_c) \tag{3.31}$$

Assuming that V_{ref} is constant during a short period after application of disturbance and by linearising equation (3.31), deviation of E_{fd} with respect to the steady state value is obtained as

$$\Delta E_{fd} = \frac{K_A}{1 + sT_A}(-\Delta V_c) \tag{3.32}$$

In the time domain equation (3.32) can be written as

$$\frac{d}{dt}\Delta E_{fd} = -\frac{K_A}{T_A}\Delta V_c - \frac{1}{T_A}\Delta E_{fd} \tag{3.33}$$

From Fig. 3.5 it is seen that:

$$\Delta V_c = \frac{1}{1 + sT_R}\Delta V_t \tag{3.34}$$

In the time domain equation (3.34) can be written as

$$\frac{d}{dt}\Delta V_c = \frac{1}{T_R}(\Delta V_t - \Delta V_c) \tag{3.35}$$

In order to obtain the state-space representation of the system the state vector should be redefined. Equations (3.33) and (3.35) introduce two new state variables, namely, ΔV_c and ΔE_{fd}. However, ΔV_t is not a state variable and should be expressed in terms of other state variables. It can be shown [1] that

$$\Delta V_t = K_5 \Delta\delta + K_6 \Delta\psi_{fd} \tag{3.36}$$

where

$$K_5 = \frac{e_{d0}}{V_{t0}}[-R_a m_1 + L_l n_1 + L_{aqs} n_1] + \frac{e_{q0}}{V_{t0}}[-R_a n_1 - L_l m_1 - L'_{ads} m_1] \tag{3.37}$$

$$K_6 = \frac{e_{d0}}{V_{t0}}[-R_a m_2 + L_l n_2 + L_{aqs} n_2] + \frac{e_{q0}}{V_{t0}}\left[-R_a n_2 - L_l m_2 + L'_{ads}\left(\frac{1}{L_{fd}} - m_2\right)\right] \tag{3.38}$$

where e_{d0} and e_{q0} are calculated as follows:

$$e_{d0} = R_E i_{d0} - X_E i_{q0} + E_B \sin\delta_0 \tag{3.39}$$

$$e_{q0} = R_E i_{q0} + X_E i_{d0} + E_B \cos\delta_0 \tag{3.40}$$

i_{d0} and i_{q0} are given by equations (3.18) and (3.19), respectively.

From the previous expressions the state-space representation of the system is given by

$$\frac{d}{dt}\begin{bmatrix} \Delta\omega_r \\ \Delta\delta \\ \Delta\psi_{fd} \\ \Delta V_c \\ \Delta E_{fd} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 & 0 \\ a_{21} & 0 & 0 & 0 & 0 \\ 0 & a_{32} & a_{33} & 0 & a_{35} \\ 0 & a_{42} & a_{43} & a_{44} & 0 \\ 0 & 0 & 0 & a_{54} & a_{55} \end{bmatrix}\begin{bmatrix} \Delta\omega_r \\ \Delta\delta \\ \Delta\psi_{fd} \\ \Delta V_c \\ \Delta E_{fd} \end{bmatrix} + \begin{bmatrix} b_1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}\Delta T_m \tag{3.41}$$

where a_{11}, a_{12}, a_{13}, a_{21}, a_{32} and a_{33} are defined by equations (3.23) to (3.30). Other coefficients are given in equations (3.42) to (3.48):

$$a_{35} = b_{32} = \frac{\omega_0 R_{fd}}{L_{adu}} \tag{3.42}$$

$$a_{42} = \frac{K_5}{T_R} \tag{3.43}$$

$$a_{43} = \frac{K_6}{T_R} \tag{3.44}$$

$$a_{44} = -\frac{1}{T_R} \tag{3.45}$$

$$a_{54} = -\frac{K_A}{T_A} \tag{3.46}$$

$$a_{55} = -\frac{1}{T_A} \tag{3.47}$$

$$b_1 = b_{11} = \frac{1}{2H} \tag{3.48}$$

3.4.4 Block diagram representation of the linearised system

Laplace transformation of equation (3.3) gives:

$$\Delta\omega_r = \frac{1}{2Hs + K_D}(\Delta T_m - \Delta T_e) \tag{3.49}$$

The variation of ψ_{fd} is determined by the field circuit dynamic equation, which is given by equation (3.41) as

$$\frac{d}{dt}\Delta\psi_{fd} = a_{32}\Delta\delta + a_{33}\Delta\psi_{fd} + a_{35}\Delta E_{fd} \tag{3.50}$$

Laplace transform calculation of equation (3.50), after some straightforward manipulation, will result in

$$\Delta\psi_{fd} = \frac{K_3}{1 + sT_3}[\Delta E_{fd} - K_4\Delta\delta] \tag{3.51}$$

where

$$K_3 = -\frac{a_{35}}{a_{33}} \tag{3.52}$$

$$K_4 = -\frac{a_{32}}{a_{35}} \tag{3.53}$$

$$T_3 = -\frac{1}{a_{33}} \tag{3.54}$$

Using equations (3.49), (3.51), (3.36), and (3.13) and using Fig. 3.5 the block diagram representation of the system including excitation system can be derived [1] and is as shown in Fig. 3.6. This block diagram will be used in small-signal stability

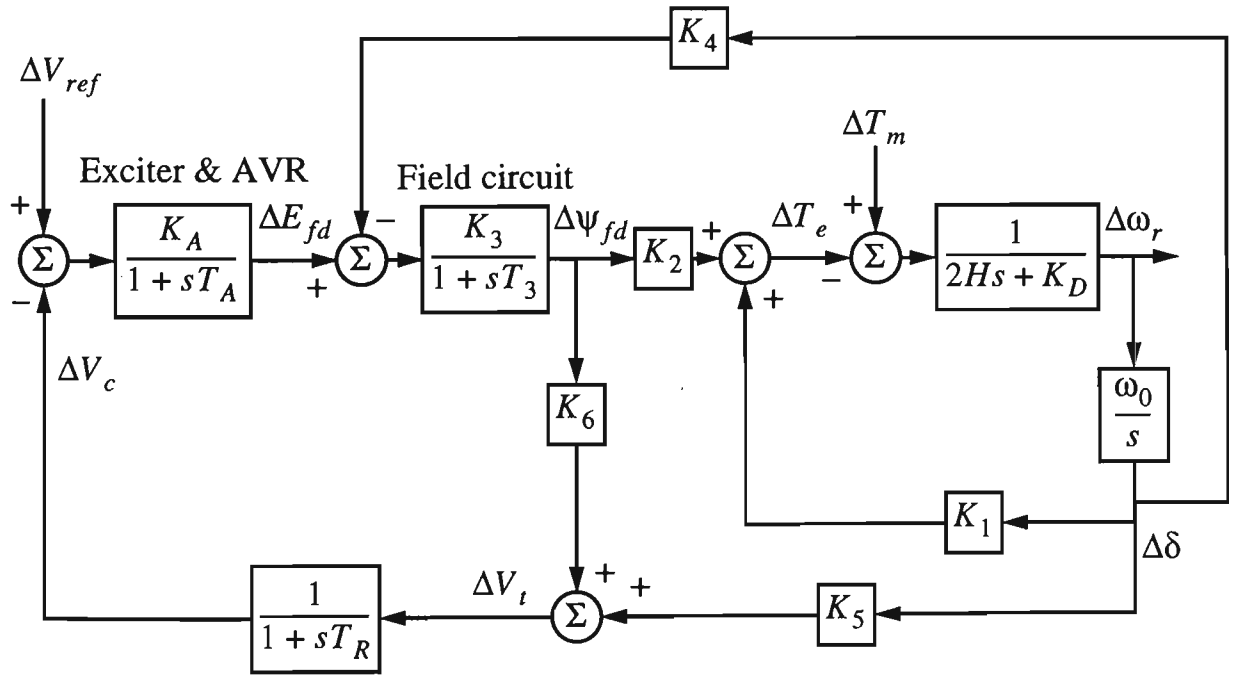


Fig. 3.6 Block diagram representation of the linearised system

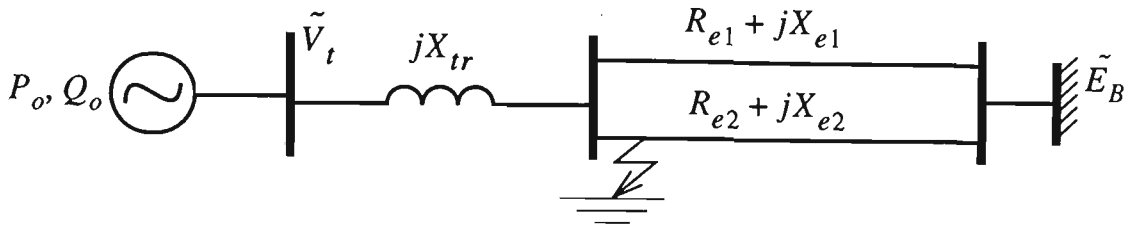


Fig. 3.7 Power system configuration for a transient stability study

studies.

3.5 Power System Model for Transient Stability

For transient stability analysis, a power system model consisting of a synchronous machine connected to a constant voltage bus through a pair of three phase transmission lines is used in simulation studies. A simplified schematic diagram of the model is shown in Fig. 3.7. A three phase transformer is used between the synchronous machine and the transmission lines to boost the machine voltage.

It is assumed that the generator is represented by a model with one d-axis and two q-axis amortisseurs. For salient pole rotors amortisseurs are damper windings which consist of copper or brass rods embedded in the pole face. Steam and gas turbines often do not have special damper windings, but their solid steel rotor offers paths for eddy currents which have effects equivalent to amortisseur currents. The machine under study is driven by a steam turbine. For the purposes of analysis, the currents in the amortisseurs may be assumed to flow in two sets of closed circuits: one set whose flux is in line with that of the field along the d-axis and the other set whose flux is along the q-axis. For system stability studies, it is seldom necessary to represent more than two or three amortisseur circuits in each axis. For the sake of simplicity only one amortisseur circuit is used along d-axis and two are used along q-axis.

The following is a summary of the synchronous machine equations as a set of first order differential equations, with time t in seconds, rotor angle δ in electrical radians, and all other quantities in per unit [1].

$$\frac{d}{dt}\Delta\omega_r = \frac{1}{2H}(T_m - T_e - K_D\Delta\omega_r) \tag{3.55}$$

$$\frac{d\delta}{dt} = \omega_0\Delta\omega_r \tag{3.56}$$

$$\frac{d\Psi_{1d}}{dt} = \omega_0\left(\frac{\Psi_{ad} - \Psi_{1d}}{L_{1d}}\right)R_{1d} \tag{3.57}$$

$$\frac{d\Psi_{fd}}{dt} = \omega_0\left[e_{fd} + \frac{(\Psi_{ad} - \Psi_{fd})R_{fd}}{L_{fd}}\right] \tag{3.58}$$

$$\frac{d\Psi_{2q}}{dt} = \omega_0\left(\frac{\Psi_{aq} - \Psi_{2q}}{L_{2q}}\right)R_{2q} \tag{3.59}$$

$$\frac{d\Psi_{1q}}{dt} = \omega_0\left(\frac{\Psi_{aq} - \Psi_{1q}}{L_{1q}}\right)R_{1q} \tag{3.60}$$

where

$$\Psi_{ad} = L''_{ads}\left(-i_d + \frac{\Psi_{fd}}{L_{fd}} + \frac{\Psi_{1d}}{L_{1d}}\right) \tag{3.61}$$

$$\Psi_{aq} = L''_{aqs}\left(-i_q + \frac{\Psi_{1q}}{L_{1q}} + \frac{\Psi_{2q}}{L_{2q}}\right) \tag{3.62}$$

Also

$$e_d = -R_a i_d + L''_q i_q + E''_d \tag{3.63}$$

$$e_q = -R_a i_q - L''_d i_d + E''_q \tag{3.64}$$

where

$$E''_d = -L''_{aqs}\left(\frac{\Psi_{1q}}{L_{1q}} + \frac{\Psi_{2q}}{L_{2q}}\right) \tag{3.65}$$

$$E''_q = L''_{ads}\left(\frac{\Psi_{fd}}{L_{fd}} + \frac{\Psi_{1d}}{L_{1d}}\right) \tag{3.66}$$

$$T_e = e_d i_d + e_q i_q + R_a I_t^2 \tag{3.67}$$

$$e_{fd} = \frac{R_{fd}}{L_{adu}} E_{fd} \tag{3.68}$$

where E_{fd} is the output of the exciter. H, K_D, R_{1d}, L_{1d}, R_{fd}, L_{fd}, R_{1q}, L_{1q}, R_{2q}, L_{2q}, R_a and L_{adu} are machine parameters. These parameters are calculated from the standard parameters of the synchronous machine, which are normally provided by the manufacturer. Such parameters for the machine under study are given in the Appendix A.

3.6 Conclusions

For the small-signal stability problem the differential equations of the synchronous machine are linearised for the specific operating condition under study. The block diagram representation of the system is then obtained which is suitable for small-signal stability studies for the specified operating point.

For the transient stability problem the nonlinear differential equations of the system are solved numerically to obtain system response.

Chapter 4

Conventional Power System Stabilisers

4.1 Overview of the Chapter

In this chapter, a conventional power system stabiliser is designed on the basis of the block diagram representation of the system introduced in Chapter 3. The design procedure is performed in the frequency domain.

4.2 Introduction

Conventional power system stabilisers (CPSSs) are basically designed on the basis of a linear model for the power system. The power system is first linearised around a specific operating point of the system. Then, assuming that disturbances are small such that the linear model remains valid, the CPSS is designed. Therefore, a CPSS is most useful for preserving dynamic stability of the power system. The procedure for deriving a linear model for this purpose was outlined in Chapter 3.

4.3 Conventional Power System Stabiliser Design

The basic function of a PSS is to add damping to the generator rotor oscillations by controlling its excitation using auxiliary stabilising signal. To provide damping, the stabiliser must produce a component of electrical torque in phase with the rotor speed deviation.

A simplified schematic diagram of a single-machine infinite-bus system, which illustrates the position of a PSS, is shown in Fig. 4.1. The system consists of a generating unit connected to an infinite bus through a transformer and a pair of transmission lines. An excitation system and automatic voltage regulator (AVR) are used to control the terminal voltage of the generator. An associated governor monitors the shaft frequency and controls mechanical power.

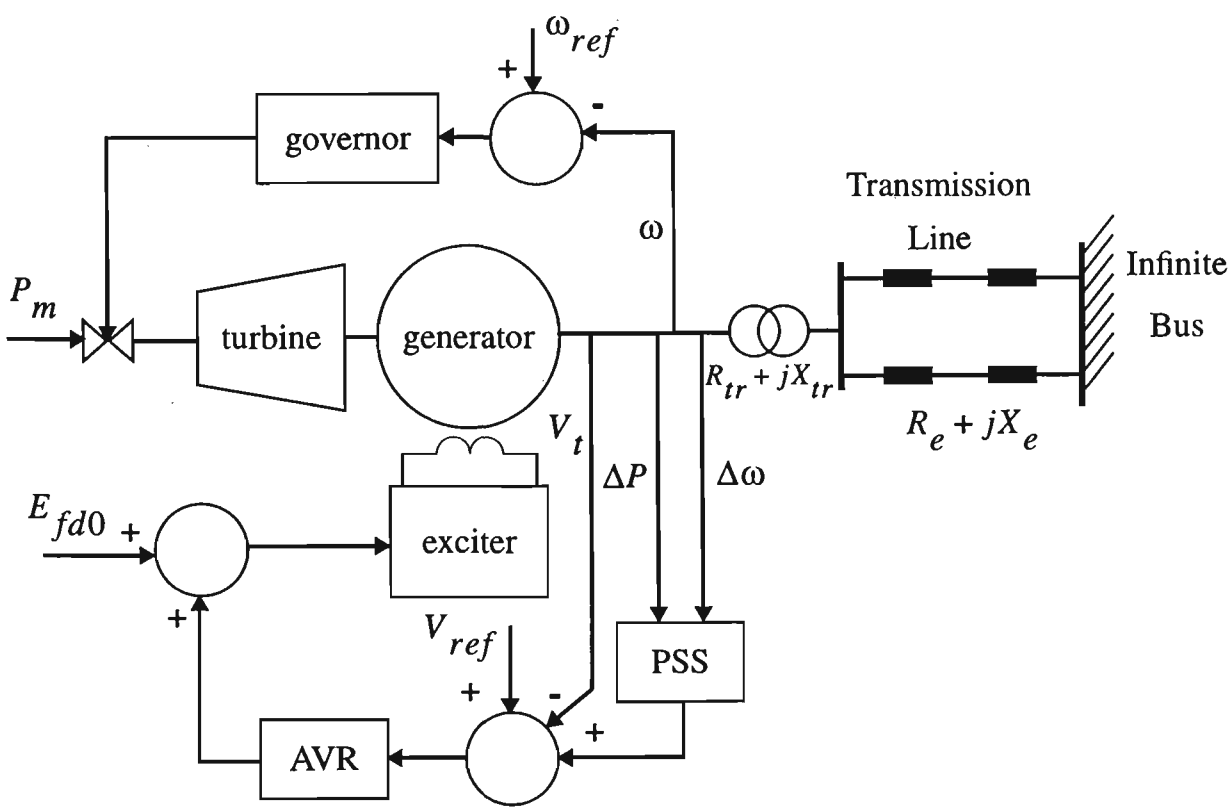


Fig. 4.1 Power system configuration

Adding a PSS to the block diagram shown in Fig. 3.6, the block diagram of the power system with PSS is obtained as shown in Fig. 4.2. Since the purpose of a PSS is to introduce a damping torque component, a logical signal to use as the input of PSS is $\Delta\omega_r$. If the exciter transfer function and the generator transfer function between ΔE_{fd} and ΔT_e were pure gains, a direct feedback of $\Delta\omega_r$ would result in a damping torque component. However, both transfer functions between ΔE_{fd} and ΔT_e exhibit frequency dependent gain and phase characteristics. Therefore, the CPSS transfer function should have an appropriate phase compensation circuit to compensate for the phase lag between the exciter input and the electrical torque. In the ideal case, with the phase characteristic of $G_{PSS}(s)$ being an exact inverse of the exciter and generator phase characteristics, the CPSS would result in a pure damping

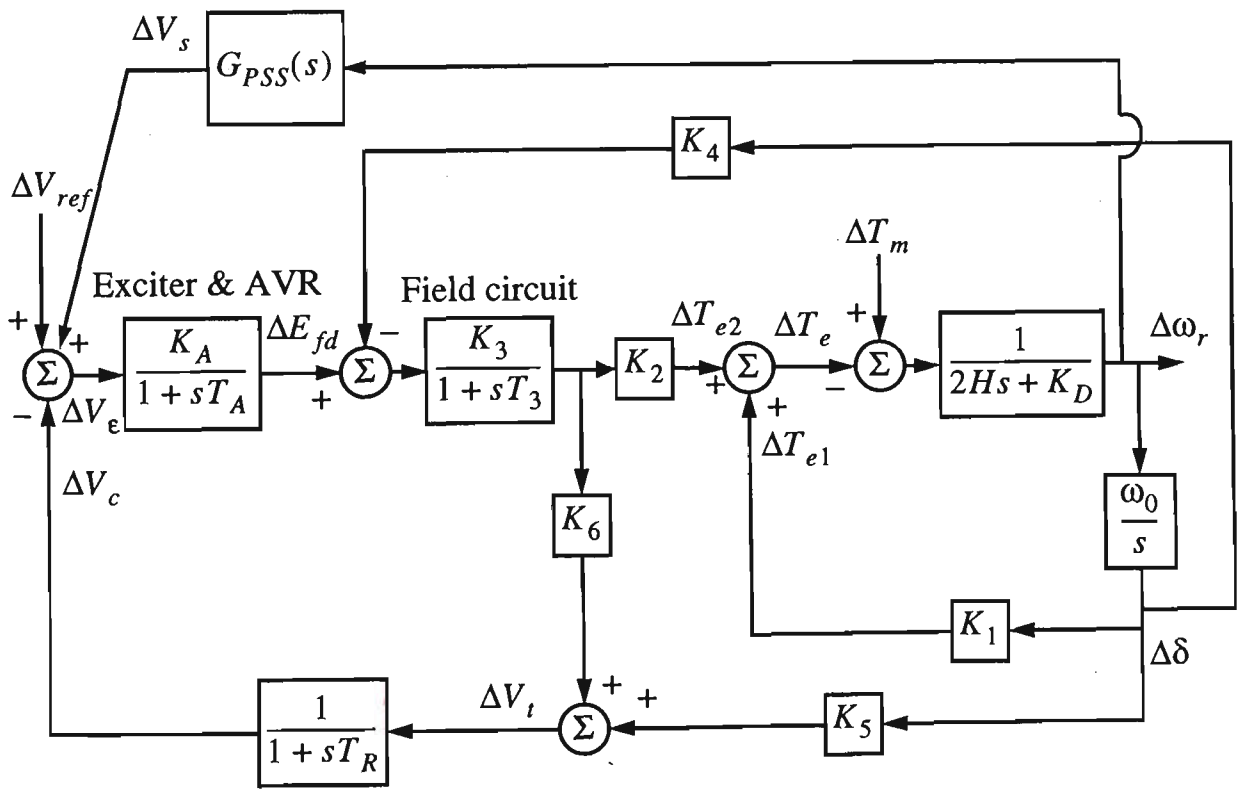


Fig. 4.2 Block diagram of a linear model of a synchronous machine with a PSS

torque at all oscillating frequencies.

The block diagram of Fig. 4.2 can be reduced to the block diagram shown in Fig. 4.3, where

$$G_1(s) = \frac{K_A K_3}{T_A T_3 s^2 + (T_A + T_3)s + 1} \tag{4.1}$$

$$G_2(s) = -\frac{K_2 K_3 s^2 + K_2 s}{2HT_3 s^3 + (2H + K_d T_3)s^2 + (K_1 T_3 \omega_0 + K_d)s + \omega_0(K_1 - K_2 K_3 K_4)} \tag{4.2}$$

$$H_1(s) = \frac{-2K_6 H s^2 - K_6 K_d s + \omega_0(K_2 K_5 - K_1 K_6)}{K_2 s} \tag{4.3}$$

$$H_2(s) = \frac{1}{1 + sT_R} \tag{4.4}$$

The block diagram in Fig. 4.3 can be further simplified as shown in Fig. 4.4,

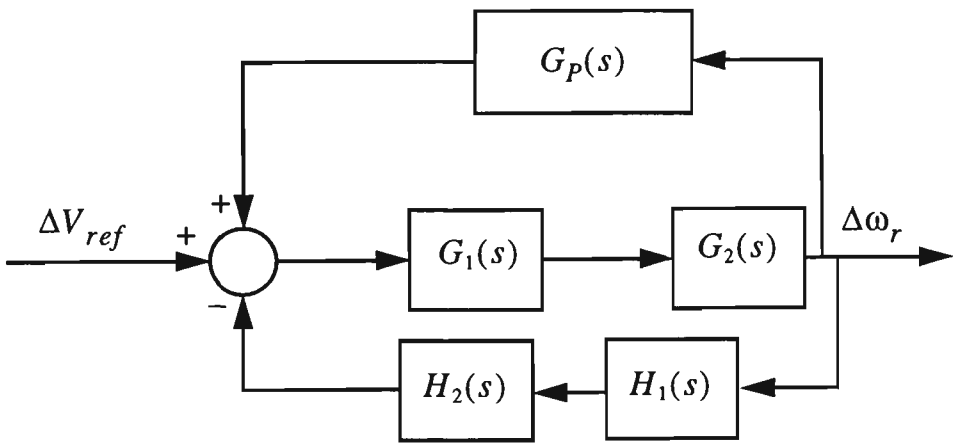


Fig. 4.3 Simplified block diagram to design a CPSS

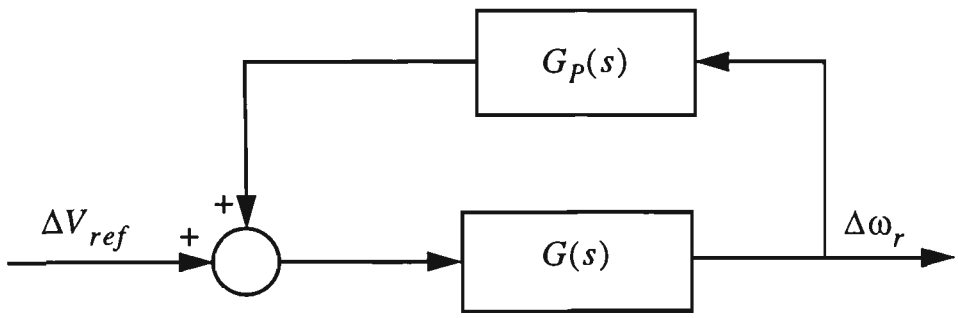


Fig. 4.4 The compact block diagram to design a CPSS

where

$$G(s) = \frac{G_1(s)G_2(s)}{1 + G_1(s)G_2(s)H_1(s)H_2(s)} \tag{4.5}$$

which is expanded to

$$G(s) = \frac{-K_A K_3(K_3 s + 1)}{a_6 s^6 + a_5 s^5 + a_4 s^4 + a_3 s^3 + a_2 s^2 + a_1 s + a_0} \tag{4.6}$$

where

$$a_6 = 2T_A T_R T_3^2 H \tag{4.7}$$

$$a_5 = T_3[2K_A T_3 H + K_A T_R(2H + K_d T_3) + 2T_R H(T_A + T_3)] \tag{4.8}$$

$$a_4 = (2H + K_d T_3)[K_A T_3 + T_R(T_A + T_3)] + 2T_3 H(T_A + T_3 + T_R) + T_A T_3 T_R(K_1 T_3 \omega_0 + K_d) \tag{4.9}$$

$$a_3 = (K_1 T_3 \omega_0 + K_d)[T_A T_3 + T_R(T_A + T_3)] + (2H + K_d T_3)(T_A + T_3 + T_R) + 2H(T_3 + K_A K_3^2 K_6) + T_A T_3 T_R \omega_0(K_1 - K_2 K_3 K_4) \tag{4.10}$$

$$a_2 = (T_A + T_3 + T_R)(K_1 T_3 \omega_0 + K_d) \tag{4.11}$$
$$+ \omega_0(K_1 - K_2 K_3 K_4)[T_A T_3 + T_R(T_A + T_3)] + K_6(2H + K_A K_d K_3^2) + 2H + K_d T_3$$

$$a_1 = \omega_0(K_1 - K_2 K_3 K_4)(T_A + T_3 + T_R) + K_A K_3^2 \omega_0(K_1 K_6 - K_5) \tag{4.12}$$
$$+ K_d(K_6 + 1) + K_1 T_3 \omega_0$$

$$a_0 = \omega_0[K_1(K_6 + 1) - K_2(K_5 + K_3 K_4)] \tag{4.13}$$

The CPSS is designed for the normal operating point, i.e., $P_o = 0.9$ pu, with a lagging power factor of 0.9 and $E_B = 1.0$ pu, where P_o is the generated active power and E_B is the infinite bus voltage. The transmission line parameters are assumed to be $R_e = 0.01$ pu and $X_e = 0.15$ pu. Frequency response method is used to design the CPSS. Bode plot of the plant without PSS ($G(s)$) for this operating condition is shown in Fig. 4.5. As shown in the figure, a resonance occurs at $\omega_r \approx 8$ rad/sec. That's why if there is a step change in V_{ref}, the rotor speed will oscillate around the synchronous speed as shown in Fig. 4.6. A PSS is required to damp the oscillations.

The CPSS is constructed of two lead stages cascaded with a wash-out term, with the following transfer function:

$$G_p(s) = K_{stab} \frac{\tau_0 s}{\tau_0 s + 1} \left(\frac{\tau_1 s + 1}{\tau_2 s + 1} \cdot \frac{\tau_3 s + 1}{\tau_4 s + 1} \right) \tag{4.14}$$

The first term in equation (4.14) is a highpass filter that is used to "wash out" the compensation effect for very low-frequency signals. It attenuates signals with angular frequency less than $(1/\tau_0)$ rad/sec. The use of this term will assure no permanent offset in the terminal voltage due to prolonged error in the power system

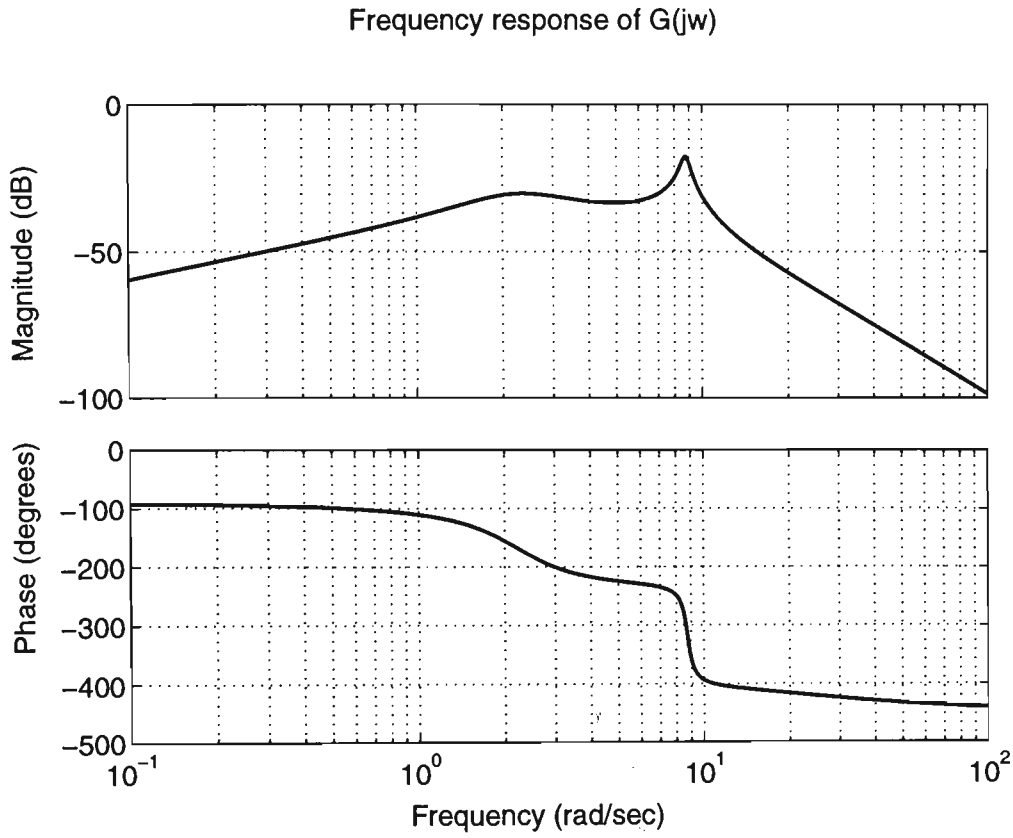


Fig. 4.5 Frequency response of the plant without PSS

frequency, such as might occur in an overload or islanding condition. The second term is a lead compensation pair that is used to compensate for the phase lag through the system.

The CPSS is a compensator in the feedback path of the system. For the case of cascade (series) compensation the effect of the controller on the closed-loop transfer function is directly determined. However, the effect of the feedback compensator on the closed-loop transfer function of the system is not easily determined. Therefore, techniques for developing feedback compensators are different and usually more involved than those for developing cascade compensators [173].

The overall transfer function of the system with the PSS is:

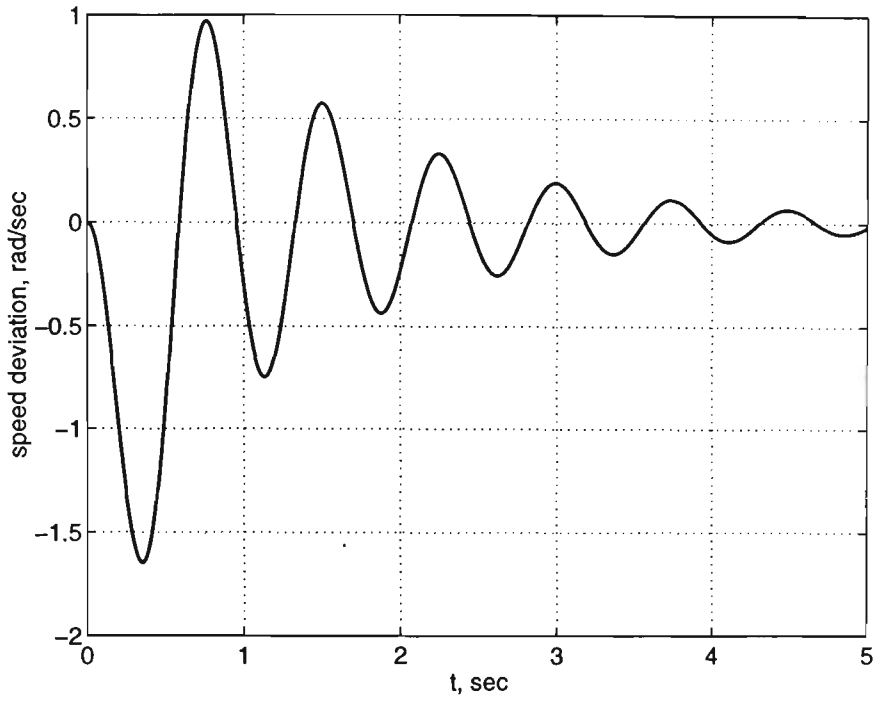


Fig. 4.6 Response of system without PSS to a 10% step change in V_{ref}

$$G_1(j\omega) = \frac{G(j\omega)}{1 - G(j\omega)G_p(j\omega)} \tag{4.15}$$

The overall transfer function can be approximated by

$$G_1(j\omega) \approx G(j\omega) \qquad \text{for } (|G(j\omega)G_p(j\omega)| \ll 1) \tag{4.16}$$

and

$$G_1(j\omega) \approx -\frac{1}{G_p(j\omega)} \qquad \text{for } (|G(j\omega)G_p(j\omega)| \gg 1) \tag{4.17}$$

The condition when $|G(j\omega)G_p(j\omega)| \approx 1$ is still undefined, in which case neither

equation (4.16) nor equation (4.17) is applicable. In the design procedure this condition is neglected. The aforementioned approximations allow investigation of the qualitative results to be obtained. After the design of $G_p(s)$ is completed, the closed-loop frequency response of the whole system will be obtained and the designer will make sure that the stabiliser performance is satisfactory in the frequency region of interest.

Thus, $G_p(s)$ should be designed such that:

❶ $|G(j\omega)G_p(j\omega)| \gg 1$ in the vicinity of the resonance region of the system. This means that the overall transfer function of the system will be approximately equal to $-\frac{1}{G_p(j\omega)}$ in the resonance region of the system. Therefore, gain and phase responses of $-\frac{1}{G_p(j\omega)}$ should be desirable in that region. The following two requirements will take care of this matter.

❷ $\frac{1}{|G_p(j\omega)|}$ is considerably less than $|G(j\omega)|$ in that region. a figure of 20 dB will be sufficient.

❸ Phase of $G(j\omega)G_p(j\omega)$ is approximately equal to *-180* degrees in that region. This means that the output of $G_p(s)$ will have an opposite phase with respect to the input to $G(s)$ causing a negative feedback (notice positive signs of the comparator inputs in Fig. 4.4). For this to happen, $G_p(s)$ must be a lead compensator.

With the above comments in mind, the CPSS is designed. The designed values are $K_{stab} = 30$, $\tau_0 = 1.6$ sec, $\tau_1 = \tau_3 = 0.16$ sec and $\tau_2 = \tau_4 = 0.01$ sec. Fig. 4.7

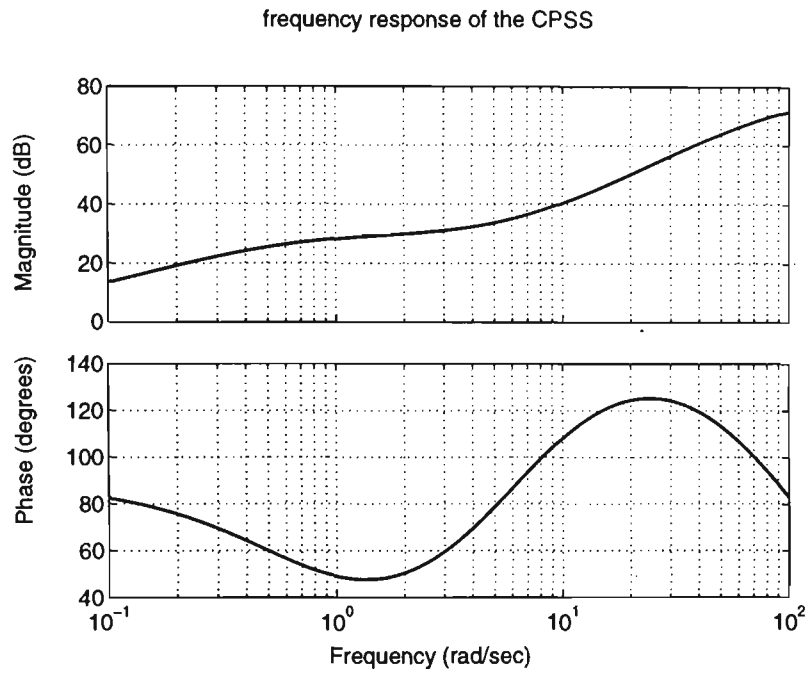


Fig. 4.7 Frequency response of the conventional PSS $(G_p(j\omega))$

shows the frequency response of this CPSS. Fig. 4.8 shows the log magnitude of $1/(G_p(j\omega))$ placed over log magnitude of $G(j\omega)$, with the aforementioned requirements achieved.

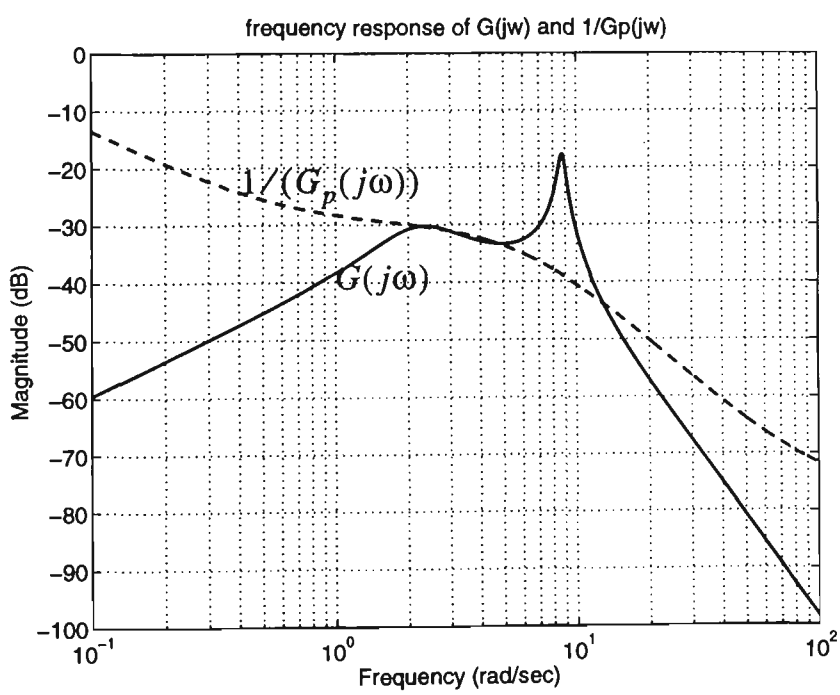


Fig. 4.8 Frequency response for log magnitude of $G(j\omega)$ and $1/(G_p(j\omega))$

By applying the designed $G_p(s)$ to the system, the closed loop frequency response will be as shown in Fig. 4.9. As is seen from the figure, the sharp resonance

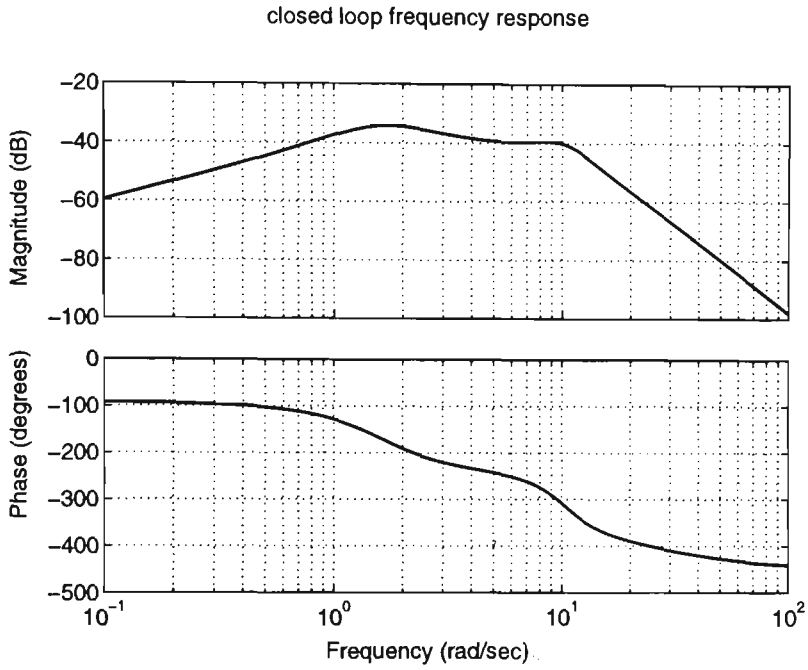


Fig. 4.9 Closed loop frequency response for the system with CPSS

which was observed in Fig. 4.5 is made smooth with a smaller magnitude and it occurs at a smaller frequency. This means that the CPSS has damped the oscillations in the time domain, as is shown in Fig. 4.10. This should be compared with the open-loop system response shown in Fig. 4.6.

4.4 Conclusions

The CPSS damps the low frequency oscillations in the shaft speed of a synchronous machine. Since the design is on the basis of a block diagram of the system derived for a specific operating point, the CPSS has the best response for this operating point. If the operating condition of the system changes, the performance of the CPSS will degrade.

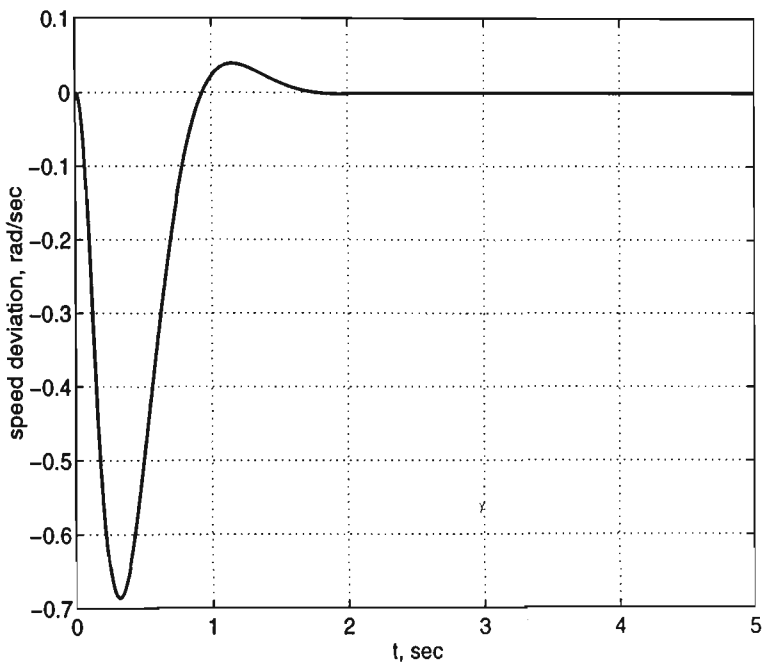


Fig. 4.10 Response of the system with CPSS to a 10% step change in V_{ref}

Chapter 5

Fuzzy Power System Stabiliser with Fixed Parameters

5.1 Overview of the Chapter

Two kinds of fuzzy logic power system stabilisers are designed in this chapter. The first is a rule-based fuzzy logic PSS which is designed according to Mamdani method for designing fuzzy logic controllers. The second is a polar fuzzy logic PSS which was first proposed by Hiyama.

5.2 Introduction

Unlike the classical control design, which requires a plant model for designing the controller, fuzzy logic incorporates an alternative way which allows control engineers to design a controller using a higher level of abstraction without having a mathematical plant model. This makes fuzzy logic controllers very attractive for ill-defined systems. With the help of fuzzy logic concepts, experts' knowledge can be used directly to implement a controller. Fuzzy logic allows one to express the knowledge with subjective concepts such as very big, too small, moderate and

slightly deviated, which are mapped to numeric ranges [142].

Fuzzy logic control implementation of power system stabilisers (PSSs) has been reported in a number of publications [96, 100, 103, 104, 174]. Due to its lower computation burden and its ability to accommodate uncertainties in the plant model, fuzzy logic power system stabilisers (FPSSs) appear to be very suitable for implementation. FPSSs can be implemented through simple microcomputers with A/D and D/A converters [101, 175, 176]. The performance of FPSSs depends on the operating conditions of the system, although it is less sensitive to changes in operating conditions than conventional linear PSSs [177].

5.3 Rule-Based Fuzzy Logic PSS

As explained in Chapter 2, a fuzzy logic controller (FLC) is a controller governed by a family of rules and a fuzzy inference mechanism. The FLC algorithm can be implemented using heuristic strategies, defined by linguistically described statements. The fuzzy logic control algorithm reflects the mechanism of control implemented by human being operators, without using any formalised knowledge about the controlled object in the form of mathematical models, and without an analytical description of the control algorithm. The main FLC processes are fuzzification, rules definition, inference mechanism and defuzzification. Thus, in a FLC the controller inputs are first fuzzified, then the proper fuzzy logic control decision is inferred based on a defined set of linguistic rules. The FLC output is then produced by defuzzifying this inferred control decision. A FPSS is a FLC which will be designed to stabilise a power system.

Fuzzification is the process of transferring the crisp input variables to the corresponding fuzzy variables. For the FPSS introduced in this chapter, speed deviation ($\Delta\omega$) and accelerating power ($\Delta P = P_m - P_e$) are fuzzified according to membership

functions as shown in Figures 5.1 and 5.2. As shown in the figures, these membership functions are normalised in the range [-1,1]. Therefore, the actual inputs, which are fed back from the power system to the FPSS, should be scaled. The speed deviation is scaled according to the relation $\Delta\omega^* = K_p \cdot \Delta\omega$ and the accelerating power is scaled according to the relation $\Delta P^* = K_d \cdot \Delta P$. $\Delta\omega^*$ and ΔP^* are the scaled variables which are inputs of the FLC. The scaling factors K_p and K_d are specified by the designer according to his knowledge of the actual ranges of $\Delta\omega$ and ΔP. Normalising the range of membership functions in the interval [-1,1] makes the design procedure easier. Usually, one of the main factors that determines the output of a FLC is the selection of the ranges of membership functions.

For each input variable seven labels are defined, namely: NB, NM, NS, ZR, PS, PM and PB which stand for negative big, negative medium, negative small, zero,

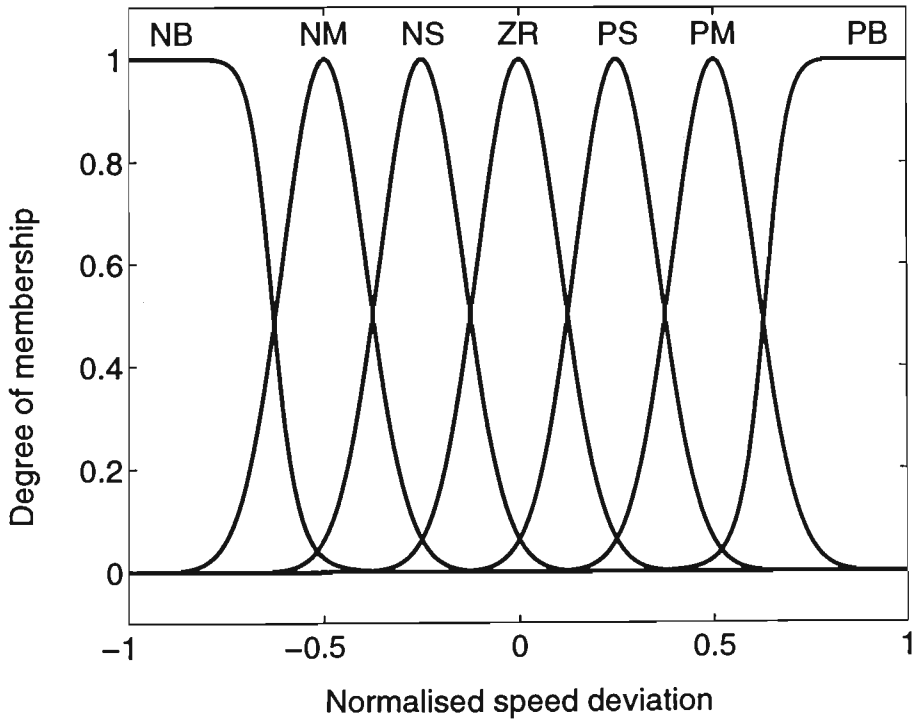


Fig. 5.1 Membership functions for input $\Delta\omega$

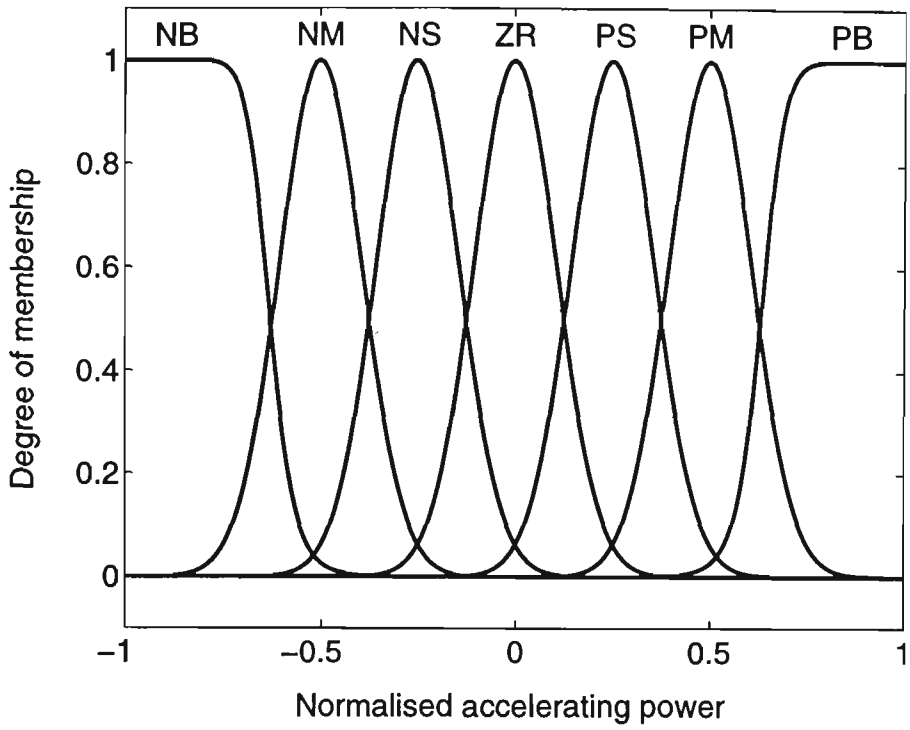


Fig. 5.2 Membership functions for input ΔP

positive small, positive medium and positive big. This means that seven membership functions are defined for each input. More membership functions will result in more smoothness in the response of the stabiliser; however, more rules are then required and this will increase the computation requirements. Usually seven membership functions are enough for the FLC construction. For this FLC, with two input variables each consisting of seven labels, a (7×7) decision table is constructed as shown in Table 5.1. Every entity in the table represents a rule, which means that 49 IF-THEN rules are used. The antecedent of each rule conjuncts $\Delta\omega$ and ΔP fuzzy set values. An example of the ith rule is:

If $\Delta\omega$ is **PS** and ΔP is **PM** then u is **PB**

which means that if the speed deviation is positive small and accelerating power is positive medium, then the output controller output should be positive big.

Table 5.1. Decision table constructed with 49 rules

$(\Delta\omega)$	(ΔP)						
	NB	NM	NS	ZR	PS	PM	PB
NB	NB	NB	NB	NS	ZR	ZR	PS
NM	NB	NB	NM	NS	ZR	PS	PM
NS	NB	NB	NM	ZR	PS	PM	PB
ZR	NB	NM	NS	ZR	PS	PM	PB
PS	NB	NM	NS	ZR	PM	PB	PB
PM	NM	NS	ZR	PS	PM	PB	PB
PB	NS	ZR	ZR	PS	PB	PB	PB

As is implied, u is the output of the FLC.

Normally, rule definition is based on the operator's experience and the engineer's knowledge. However, it has been noticed in practice that for monotonic systems a symmetrical rule table is appropriate, although sometimes it may need slight adjustment based on the behaviour of the specific system. A system is considered to be monotonic if it has symmetrical responses for negative and positive values of the inputs. If the system dynamics are not known or are highly non-linear, trial-and-error procedures and experience play an important role in defining the rules.

The crisp output of the FPSS is calculated using the concepts explained in Chapter 2. Mamdani method [131] is used as the inference mechanism. In this method, the output fuzzy set is obtained by an *and operator* between the degree of firing (DOF) and the consequent fuzzy set. The DOF of the ith rule is a scalar value which equals the minimum of the two antecedent membership degrees [178]. For example, if $\Delta\omega$ is PS with a membership degree of 0.6 and ΔP is PM with a membership degree of 0.4 then the degree of firing of this rule is 0.4. Normally, the membership functions of the consequent fuzzy sets are cut to the respective DOF; in

this way the fuzzy sets inferred by individual rules are obtained. The next step in the process is the aggregation of the individual rule outputs to obtain the overall system output. Rule aggregation is done by an *or operator* on the individual rule outputs. If maximum operation is used as the *or operator*, the maximum of the individual output fuzzy sets are obtained as the aggregated output fuzzy set. This mechanism of obtaining the output fuzzy set is called the *min-max inference* and was first used by Mamdani and Assilian [131]. Fig. 5.3 shows this procedure graphically. For this graphical representation, the lth rule has been assumed to be of the form:

IF x_1 is F_1^l and x_2 is F_2^l THEN u is G_l

Two rules are used for graphical illustration. The aggregated output fuzzy set is

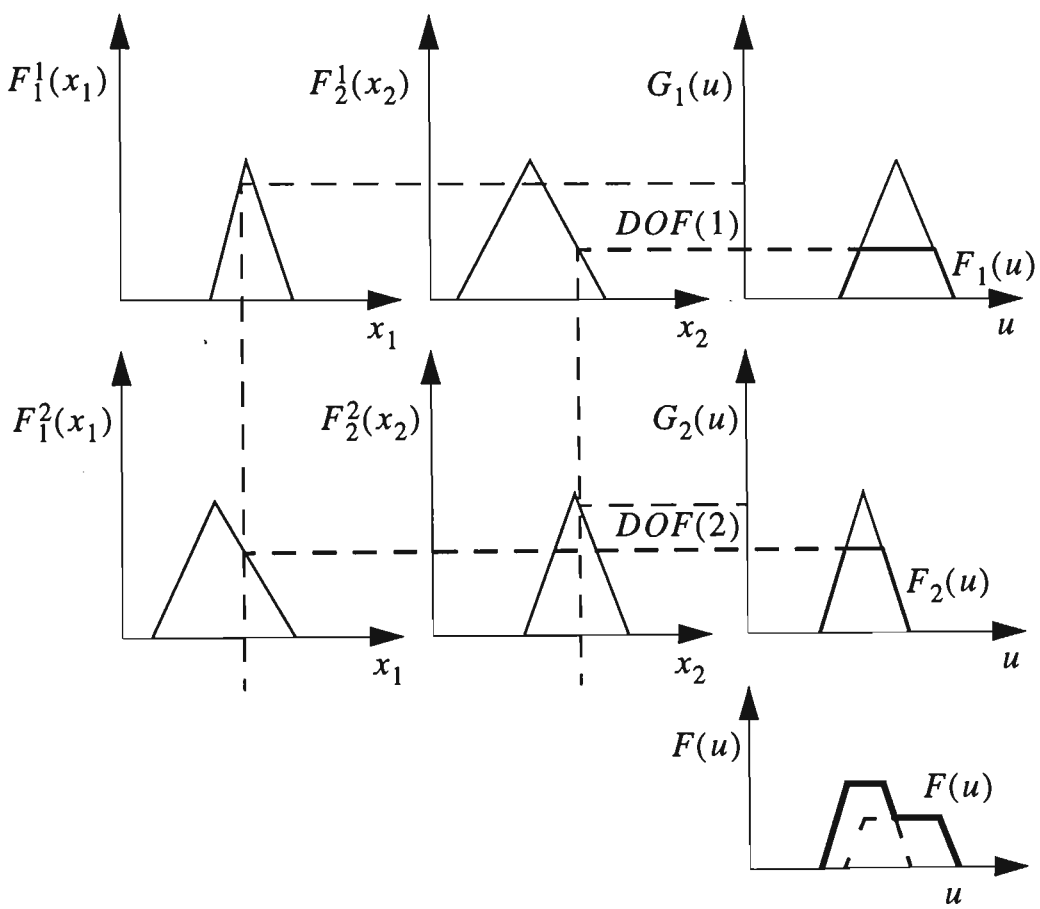


Fig. 5.3 Graphical representation of fuzzy inference mechanism

obtained to be $F(u)$. The min-max inference mechanism is simple and at the same time sufficient for most practical engineering problems.

After obtaining the output fuzzy set, it is defuzzified according to the membership functions shown in Fig. 5.4. As in the case of input membership functions, the output is also normalised. The designer will specify a scaling factor K_u, to scale the output obtained from the FPSS to a value required by the power system, according to the relation $\Delta u^* = K_u \cdot u$. Centre of area method is used for defuzzification.

5.3.1 Simulation results

As in Chapter 4, the FPSS is designed for the normal operating point, i.e., $P_o = 0.9$ pu with a power factor of 0.9 lagging and $E_B = 1.0$ pu. The transmission

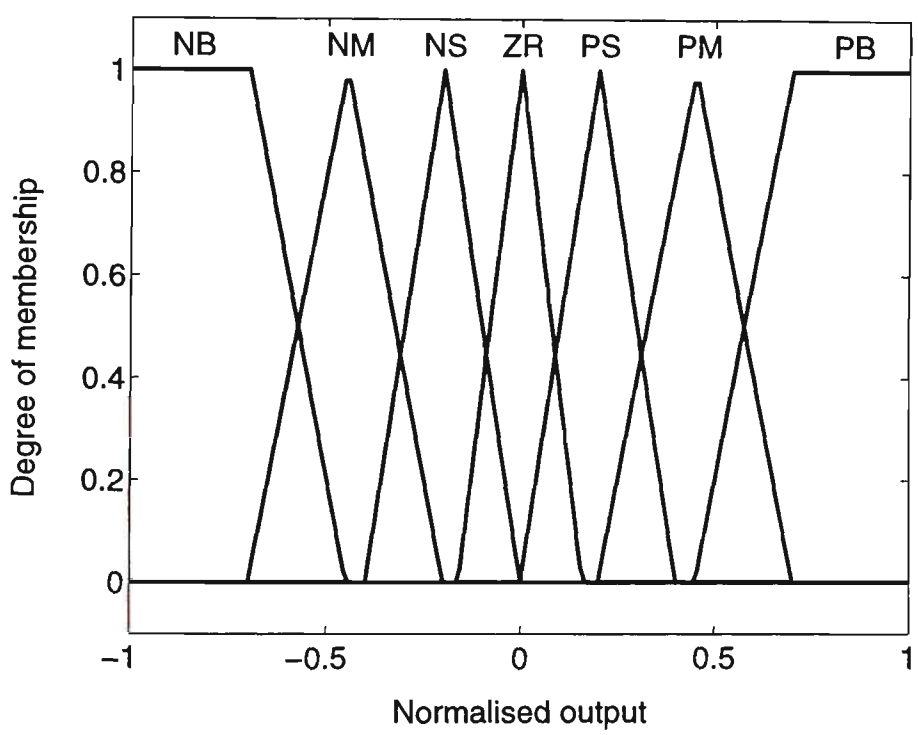


Fig. 5.4 Membership functions for output u

line parameters are assumed to be $R_e = 0.03$ pu and $X_e = 0.15$ pu. A performance index J_p is defined as shown in the following equation:

$$J_p = \sum_{k=0}^{n} |\Delta\omega(k)| \cdot t_k \tag{5.1}$$

where $\Delta\omega(k)$ = speed deviation at the kth sample

$t_k = k\Delta T = kth$ sampling time after disturbance is applied

ΔT = The sampling period

n = total number of samples for a specific time interval that simulation is performed. The optimum values of K_p, K_d and K_u are determined by minimizing the performance index J_p. For this specific operating point these optimum values are obtained to be 0.25, 0.7 and 0.5, respectively. It should be noted that the model differential equations derived in Chapter 3 or the transfer functions derived in Chapter 4 are not used in the design procedure of the FPSS. Only known linguistic rules governing the behaviour of the system are used in the design. The model differential equations are used only for simulation studies to obtain the system response. In other words, it was assumed that the parameters of the machine were not available to design a controller based on the differential equations of the power system.

System response to a step change in V_{ref} equal to 0.1 pu is shown in Fig. 5.5. Performance of this FPSS will be compared with the performance of a CPSS later in Chapter 9.

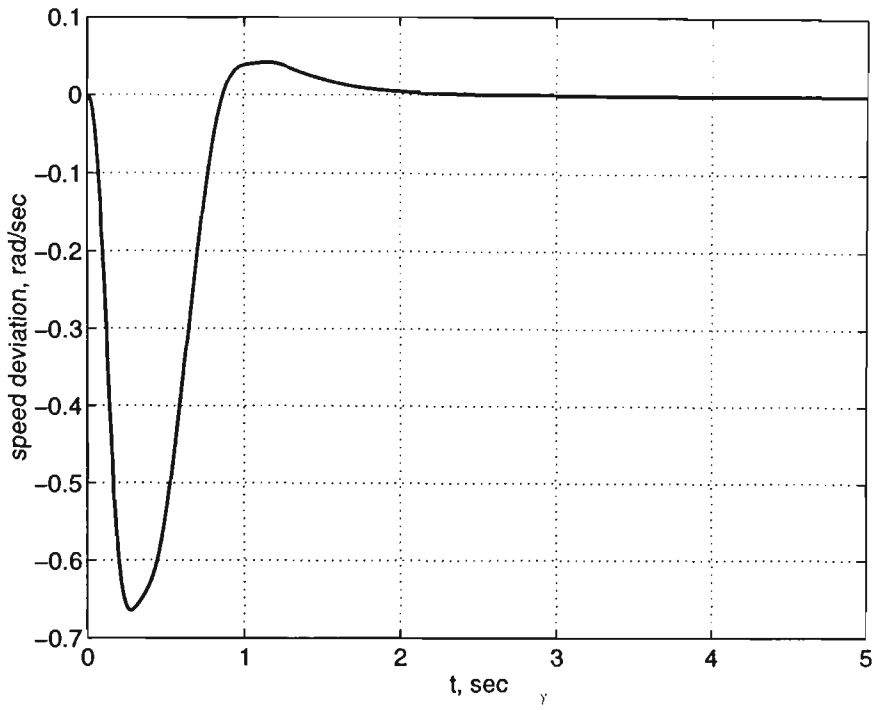


Fig. 5.5 Response of the machine with a rule-based FPSS to a 10% step change in V_{ref}

5.4 Polar Fuzzy PSS

This type of FPSS was first proposed by Hiyama [179]. The scheme is based on defining the state of the synchronous machine in the speed/acceleration phase plane as shown in Fig. 5.6. The point *P(k)* is given at the kth sampling by

$$P(k) = (Z_s(k), A_s(k)) \tag{5.2}$$

where $Z_s(k)$ is speed information and $A_s(k)$ represents scaled acceleration information.

On the basis of the speed/acceleration state of the synchronous machine and a set of simple fuzzy logic control rules, the desired stabilising signal is generated to shift the generator state to the origin of phase plane, i.e. the equilibrium point. In

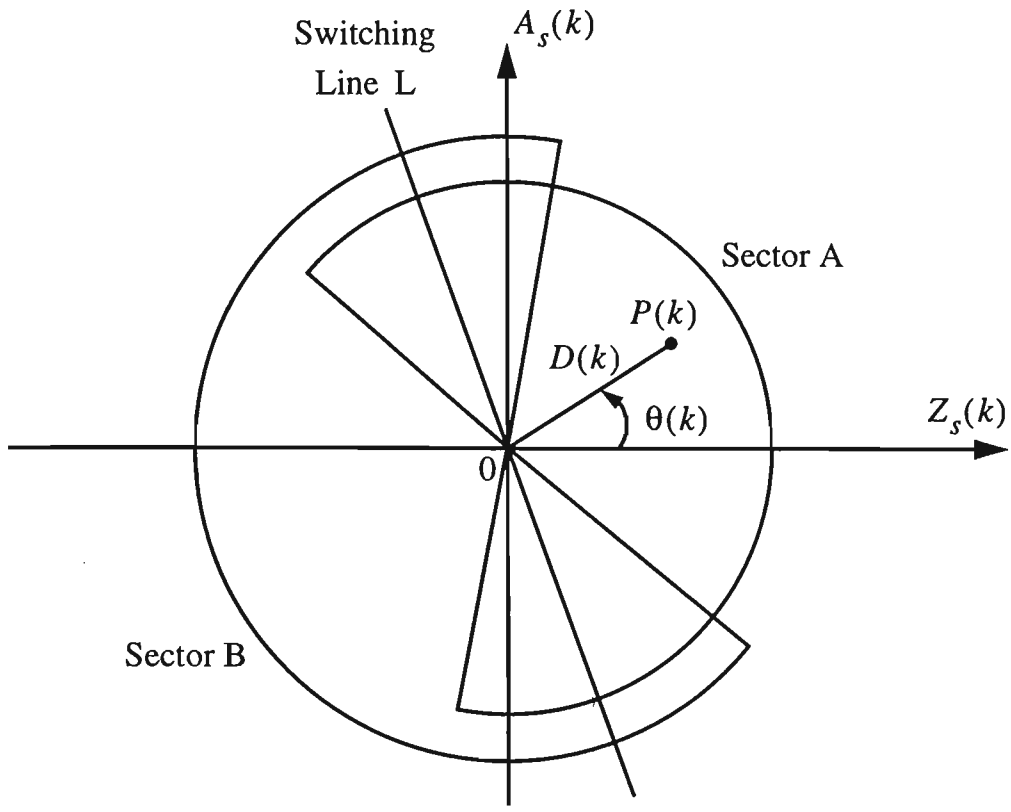


Fig. 5.6 Speed/Acceleration phase plane

order to satisfy the required stabiliser functions, two membership functions $N(\theta(k))$ and $P(\theta(k))$ are defined as shown in Fig. 5.7 to represent sectors A and B respectively. The values of $N(\theta(k))$ and $P(\theta(k))$ give the grades of deceleration and acceleration controls at the kth sampling, respectively. The term α gives the size of cross section between sectors A and B. By using these two membership functions, the stabilising signal $u(k)$ is calculated, as is given in the following equation. The distance $D(k)$ and the phase $\theta(k)$ are determined from the current state of the machine.

$$u(k) = G(k)\frac{N(\theta(k)) - P(\theta(k))}{N(\theta(k)) + P(\theta(k))} \cdot U_{max} \tag{5.3}$$

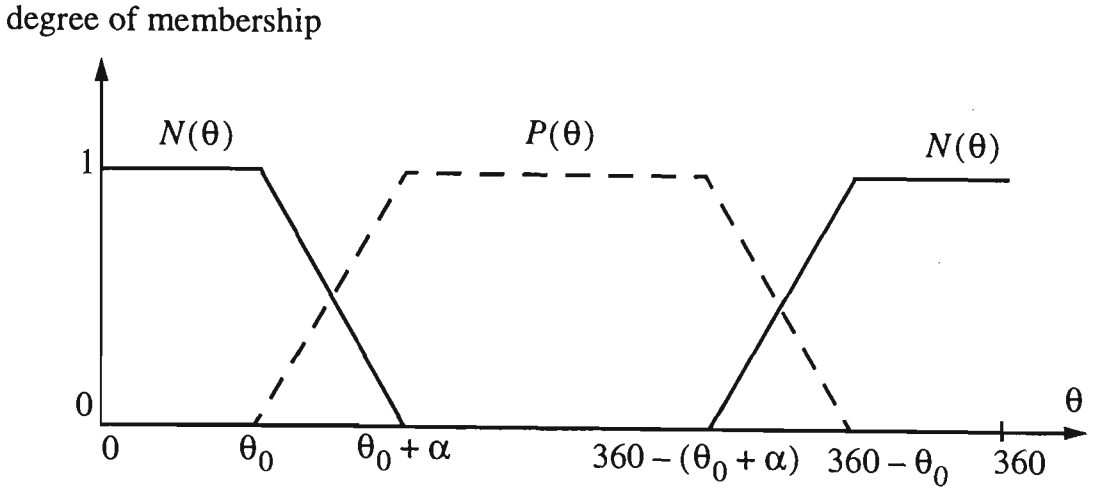


Fig. 5.7 Fuzzy membership functions for the polar FPSS

where

$$G(k) = \begin{cases} \dfrac{D(k)}{D_r} & \text{for } D(k) \le D_r \\ 1 & \text{for } D(k) > D_r \end{cases} \tag{5.4}$$

$$D(k) = \sqrt{Z_s(k)^2 + A_s(k)^2} \tag{5.5}$$

$$\theta(k) = \text{atan}\left(\frac{A_s(k)}{Z_s(k)}\right) \tag{5.6}$$

In the above equations, $G(k)$ is the stabiliser gain related to the distance $D(k)$ from the equilibrium, as is shown in Fig. 5.6. The term U_{max}, i.e. the maximum size of the stabilising signal, the term D_r specifying the gain $G(k)$, and the term α indicating the size of cross sections, are the adjustable parameters of the polar FPSS.

The algorithm to construct this FPSS is as follows:

(a) Sample the generator speed deviation $\Delta\omega(k)$ and call it $Z_s(k)$.

(b) Sample the accelerating power of the generator and scale it according to the equation

$$A_s(k) = K_d \Delta P(k) \tag{5.7}$$

where K_d is the acceleration scaling factor.

(c) Calculate state variables $\theta(k)$ and $D(k)$ according to Fig. 5.6.

(d) Evaluate fuzzy membership functions $N(\theta(k))$ and $P(\theta(k))$ according to Fig. 5.7.

(e) Evaluate the control gain $G(k)$.

(f) Calculate control signal $u(k)$ according to (5.3).

(g) Return back to step (a) for next sample $k+1$.

5.4.1 Simulation results

As in the previous section, the polar FPSS is designed for the normal operating point, i.e., $P_o = 0.9$ with power factor of 0.9 lagging and $E_B = 1.0$ pu. The transmission line parameters are assumed to be $R_e = 0.03$ pu and $X_e = 0.2$ pu. The optimum values of U_{max}, K_d, D_r, and α are determined such that the performance index J_p defined in equation (5.1) is minimised. For this specific operating point these optimum values are obtained to be 0.5, 7.5, 5.2, and 90 degrees, respectively. System response to a step change in V_{ref} equal to 0.1 pu is shown in Fig. 5.8. The phase

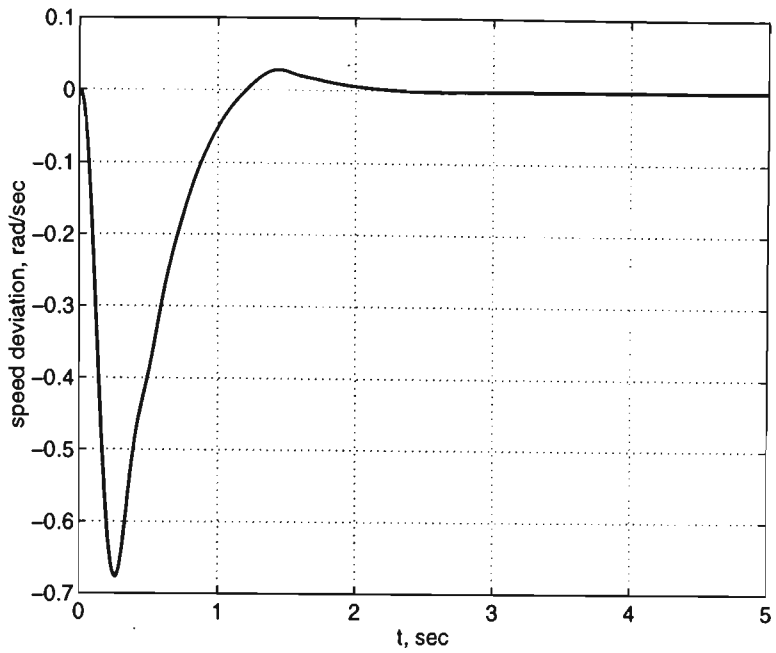


Fig. 5.8 Response of the machine with a polar FPSS to a 10% step change in V_{ref}

trajectory of the system after applying the disturbance is shown in Fig. 5.9.

Speed deviation responses for the CPSS, rule-based FPSS and polar FPSS are compared in Fig. 5.10 for normal operating condition. In this case the responses are similar because all PSSs were designed for this operating point. In order to see how the performances of the PSSs are when the operating condition changes, the same disturbance (a step change in V_{ref} equal to 0.1 pu) has been applied for the operating condition P_o = 0.7 with power factor of 0.85 lagging and E_B = 1.0 pu. The transmission line parameters are assumed to be R_e = 0.03 pu and X_e = 0.4 pu. Speed deviation responses for a step change in V_{ref} equal to 0.1 pu are shown in Fig. 5.11. As seen from the figure, the CPSS degrades more due to the changes in the operating condition, compared to the FPSSs. More comparison studies will be done in Chapter 9.

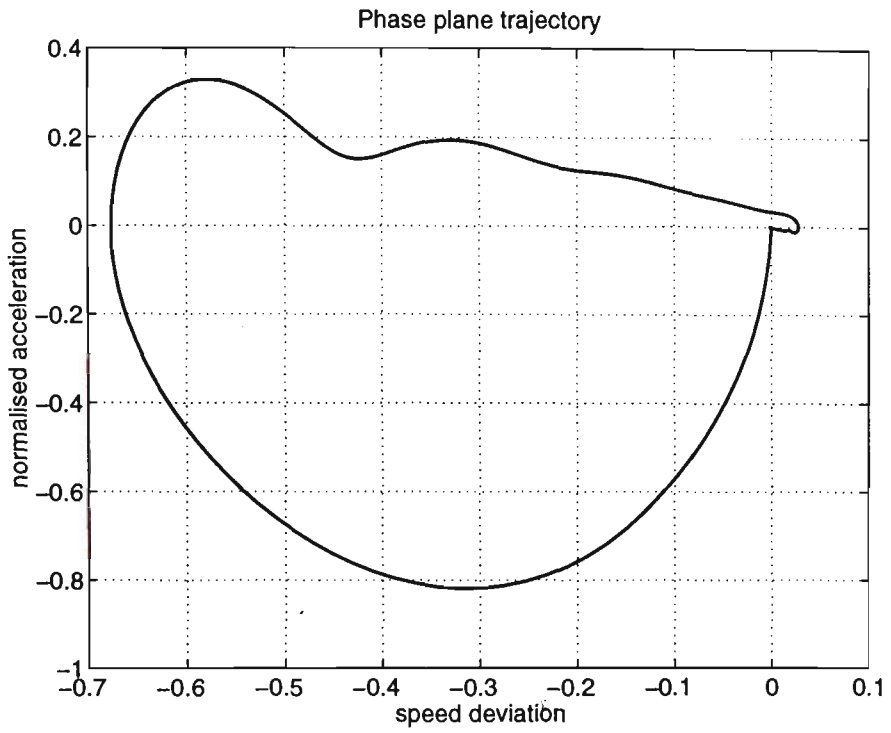


Fig. 5.9 Phase trajectory of the system with the polar FPSS after applying a disturbance

5.5 Conclusions

Two kinds of fuzzy logic PSSs were proposed in this chapter. The responses of both FPSSs are satisfactory. It was shown that if the operating condition of the power system changes, there will be less degradation in the responses of FPSSs compared to the response of a CPSS.

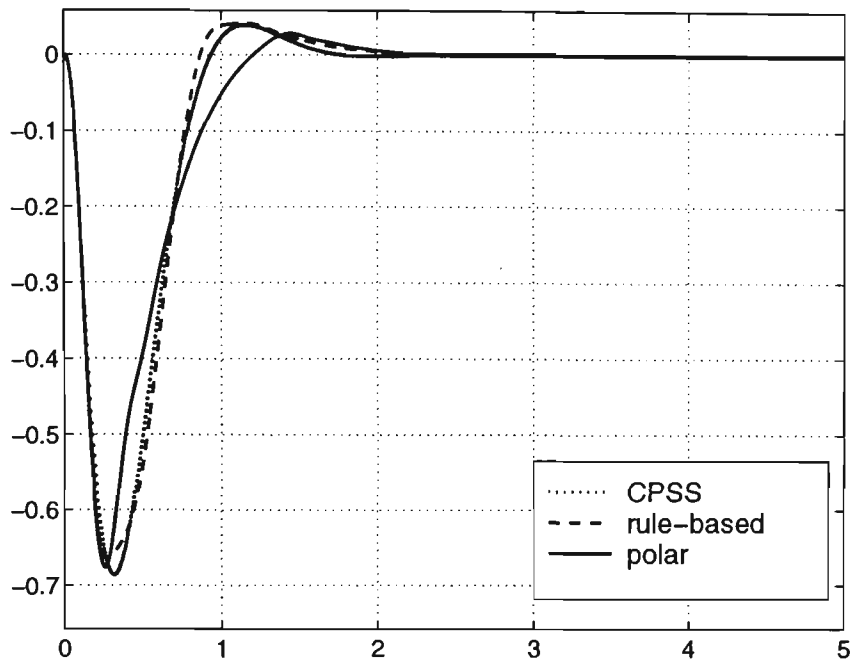


Fig. 5.10 Comparison of the responses of the CPSS, rule-base FPSS and polar FPSS after applying a disturbance for the normal operating condition

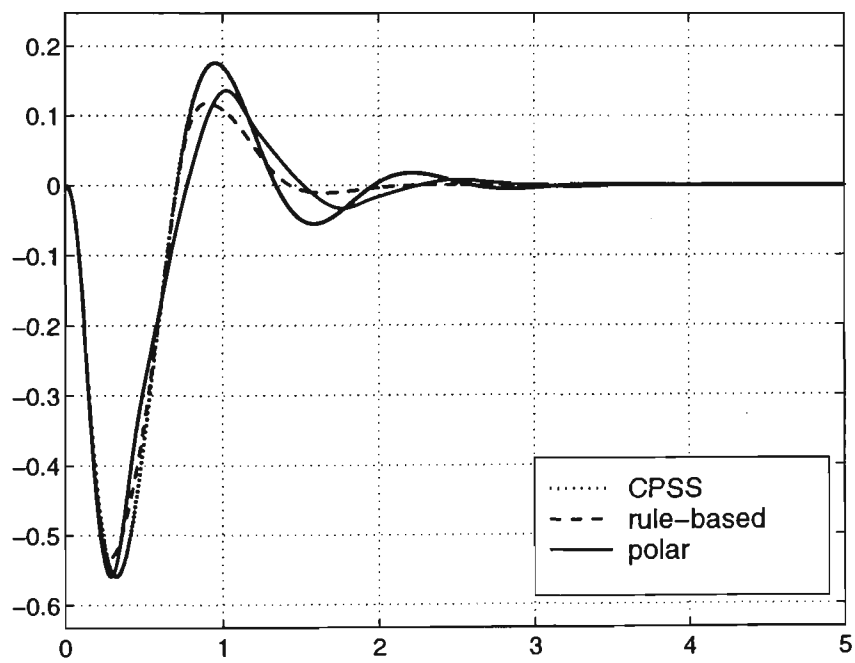


Fig. 5.11 Comparison of the responses of the CPSS, rule-base FPSS and polar FPSS after applying a disturbance when the operating condition is changed

Chapter 6

On-Line Tuning of the Fuzzy Power System Stabiliser

6.1 Overview of the Chapter

In this chapter, two schemes for on-line tuning of the FPSS are proposed. The first scheme uses the artificial neural networks to tune the FPSS. The second scheme employs a fuzzy logic system as the tuner for the FPSS.

6.2 Introduction

A rule-based fuzzy logic power system stabilizer (FPSS) was developed in Chapter 5 using speed deviation and accelerating power as the controller input variables. The inference mechanism of the fuzzy logic controller was represented by a (7×7) decision table, i.e. 49 IF-THEN rules. Two scaling factors K_p and K_d were introduced to scale $\Delta\omega$ and ΔP, respectively.

Although the design of a FPSS is based on some linguistic rules, which are normally derived by the experts, the performance of the FPSS depends on different

parameters of the fuzzy system. One of the important parameters of the FLS is the range of inputs, which can be changed by the scaling factors. A set of scaling factors suitable for a specific configuration of the power system and a specific operating condition may not be suitable for another configuration and/or operating condition. A better performance can be achieved if K_p and K_d are changed appropriately for the new configuration. In this chapter two different schemes are proposed for tuning the FPSS.

In the first scheme an artificial neural network (ANN) is used to tune the FPSS. The scaling factors are the outputs of an ANN which gets the operating conditions of the power system as inputs. In the second scheme a FLS is used to do the same job.

These mechanisms of tuning the FPSS make the FPSS adaptive to changes in the operating conditions. Therefore, the degradation of the system response, under a wide range of operating conditions, is less compared to the system response with a fixed-parameter FPSS.

6.3 On-Line Tuning of the FPSS using Neural Networks

6.3.1 Introduction

The recent growth in attention to ANNs has led to suggestions for the combined use of fuzzy logic and ANNs in intelligent control [180, 181]. Similarities exist between the ANNs and the fuzzy logic controllers (FLC). Both techniques are free from the *true/false* restriction of conventional logic systems. In a multi-layer ANN of feedforward type, input nodes record the features and pass activation values to the output layer through a hidden layer. The addition of the hidden layers to the two-layer perceptron networks allows these networks to represent any continuous mapping from input to output. The back-propagation method of Rumelhart et. al.

[182], or any other appropriate training technique, adjusts the connection weights of the network to improve the match between the output of the network and the correct results.

Hybrid PSSs using fuzzy logic and ANNs have been reported in the literature. Hiyama applied an ANN for real time tuning of a FPSS [120]. Sharaf and Lie [183] used a fuzzy logic gain scheduler to tune a neural-network-based PSS. It was shown that both schemes were effective in enhancing power system stability.

In this chapter the parameters of a rule-based FPSS are tuned with an ANN, making it adaptable to changes in the operating conditions. It is then applied to a mathematical model of a synchronous machine. Responses of the machine subjected to a fault in the transmission line is obtained by nonlinear simulations. System responses with the neural-network-tuned fuzzy logic power system stabilizer (NFPSS) for three different operating conditions are then compared with a fixed-parameters FPSS and a conventional power system stabilizer (CPSS).

6.3.2 A brief background on artificial neural networks

The basic processing element of an ANN is often called a neuron, by analogy with neurophysiology. Neurons perform as summing and nonlinear mapping junctions. In some cases they can be considered as threshold units that fire when their total input exceeds certain bias levels. Neurons usually operate in parallel and are often organised in layers. Each connection strength is expressed by a numerical value called a weight [184].

An elementary feedforward neural network (FNN) architecture of m neurons receiving n inputs is shown in Fig. 6.1. Its output and input vectors are, respectively

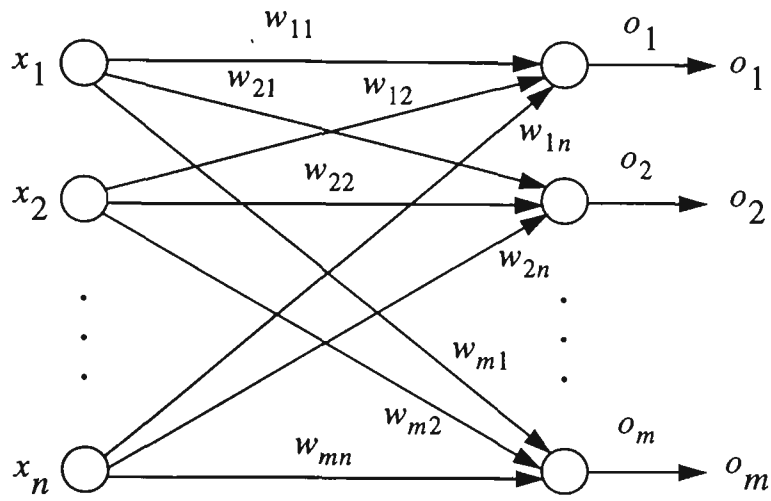


Fig. 6.1 Single layer feedforward neural network

$$O = [o_1, o_2, \ldots, o_m]^T$$
$$X = [x_1, x_2, \ldots, x_n]^T \tag{6.1}$$

Weight w_{ij} connects the *ith* neuron with the *jth* input. The activation value for the *ith* neuron can be written as

$$net_i = \sum_{j=1}^{n} w_{ij} x_j \qquad \text{for i = 1, 2, ..., m} \tag{6.2}$$

The *ith* output can be calculated as

$$o_i = f(W_i^T X) \qquad \text{for i = 1, 2, ..., m} \tag{6.3}$$

where $f(net_i)$ is called the activation function and can be any nonlinear function.

Also,

$$W_i = [w_{i1}, w_{i2}, \ldots, w_{in}] \tag{6.4}$$

This type of network can be connected in cascade to create a multi-layer FNN. In such a network, the output of a layer is the input to the following layer. Even though the FNN has no explicit feedback connection, the output values are often compared with the desired output values, provided by a "supervisor". The error between the outputs of the ANN and the desired values can be employed for adapting the network's weights. The error is used to modify weights so that the error decreases. This type of learning is called *supervisory learning.* A set of input and output patterns called a t*raining set* is required for this learning mode.

6.3.3 Tuning scheme

In order to tune the FPSS, speed deviation is scaled according to the relation $\Delta\omega^* = K_p \cdot \Delta\omega$ and accelerating power is scaled according to the relation $\Delta P^* = K_d \cdot \Delta P$. Also, the output of the FPSS is scaled according to the relation $\Delta u^* = K_u \cdot u$. The scaling factor for the output u is fixed to a suitable number, i.e. 0.5 for the system under study. This number should be chosen to be greater than the required saturation level of the PSS. The FPSS is tuned by computing optimum K_p and K_d, exploiting an ANN as shown in Fig. 6.2. The evaluation of the optimality is checked by the discrete time performance index J_p as shown in the following equation:

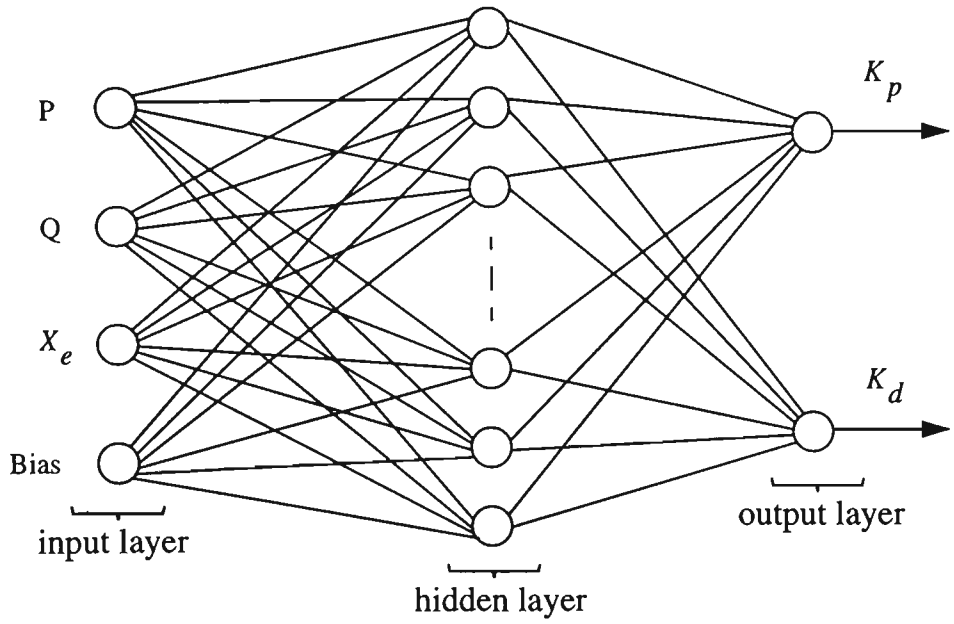


Fig. 6.2 Neural network tuner configuration

$$J_p = \sum_{k=0}^{n} |\Delta\omega(k)| \cdot t_k \tag{6.5}$$

where $\Delta\omega(k)$ = speed deviation at the kth sample

$t_k = k\Delta T = kth$ sampling time after fault occurrence

ΔT = The sampling period

n = total number of samples for a specific time interval that simulation is performed.

The ANN is composed of three layers, i.e., an input layer, a hidden layer, and an output layer. The generated active power P_o and reactive power Q_o are selected for input signals to represent the operating condition of the synchronous machine. X_e, the total reactance of the transmission line, is also selected as an input to

represent the external impedance information. The activation functions for the hidden layer are sigmoid functions, i.e. $f(x) = \frac{1-e^{-x}}{1+e^{-x}}$ and the output characteristics are linear functions. Sigmoid transfer function is commonly used in backpropagation networks, in part because it is differentiable. In order to reduce the required computation time in the learning process a bias signal is also used as one of the inputs of the ANN.

For various sets of input data to the ANN, the optimum values of K_p and K_d are searched sequentially using nonlinear simulations when one end of the transmission line is subjected to a three-phase to ground fault. The duration time of the fault is set to *40 msec*.

The input data to the ANN is a combination of the values shown in Table 6.1. From various combinations of the input data, the optimal parameters of the FPSS are searched. Then a set of learning data is composed for training the ANN. A total of 60

Table 6.1. Input signals to the neural network

P_o (pu)	0.4	0.6	0.8	1.0
Q_o (pu)	-0.3	0	0.3	0.6
X_e (pu)	0.1	0.2	0.3	0.4

learning data are prepared, as shown in Appendix B. Since a wide range of operating conditions is used to prepare the training set, this set will cover the operating conditions usually encountered by the power system. On the other hand, this training set cannot be very big. The reason is that in practice the power system should be

tested for every operating point in this set to adjust the scaling factors. If the training set is too big, these tests will be unfeasible.

6.3.3.1 Training the neural network

An optimisation technique called Levenberg-Marquardt method is used to train the ANN [185]. This method is more powerful than gradient descent (backpropagation training method), but requires more memory of the computer. Since the training process is done off-line this requirement does not degrade the performance of the system. This means that after preparing the training set by performing practical tests, the FNN will be trained before applying it to the power system. The Levenberg-Marquardt update rule is

$$\Delta W = (J^T J + \mu I)^{-1} J^T e \tag{6.6}$$

where J is the Jacobian matrix of derivatives of each error to each weight, μ is a scalar quantity, and e is the error vector. If μ is very large, the above expression approximates gradient descent, while if it is small then it becomes the Gauss-Newton method. Since the Gauss-Newton method is faster, but tends to be less accurate when near an error minimum, μ is adjusted during training. Fig. 6.3 shows the variation of error during the training process.

To show the efficiency of the Levenberg-Marquardt training algorithm, it is compared with the back-propagation algorithm. Fig. 6.4 shows the variation of error during the training process when the back-propagation algorithm is used. On comparing Figures 6.3 and 6.4, it is observed that the Levenberg-Marquardt algorithm converges faster. Also, with the back-propagation algorithm the possibility

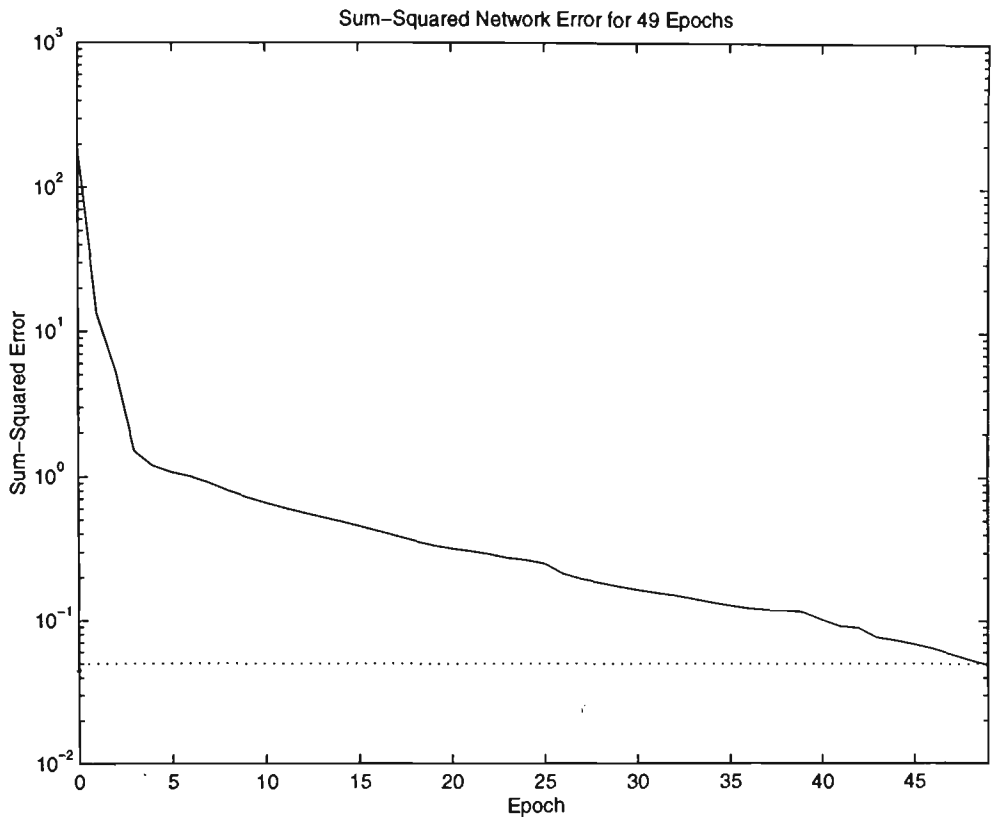


Fig. 6.3 Variation of error during the training process using the Levenberg-Marquardt algorithm

of instability in the training process is more, i.e., the error may increase or have an oscillatory behaviour.

6.3.4 Simulation results

In order to show the effectiveness of the tuning scheme of the FPSS by an ANN, the responses of the fixed-parameters FPSS and NFPSS for two different operating conditions are shown in this section. The speed deviation responses of the FPSS and NFPSS when the power system is subjected to a three-phase to ground fault occurring on one of the transmission lines (Fig. 3.7) are obtained using computer simulations. The nonlinear model for transient stability studies is used as explained in Section 3.5. It is assumed that the fault occurs at t = 0.2 sec with a

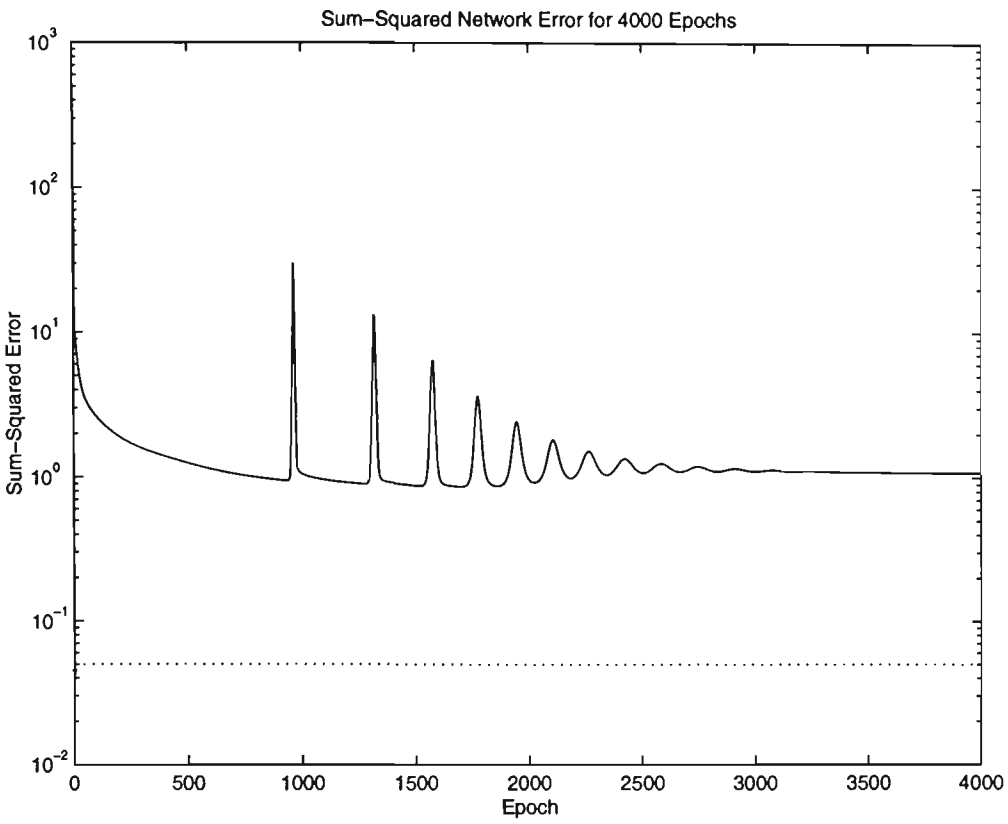


Fig. 6.4 Variation of error during the training process using the back-propagation algorithm

duration of 40 msec after which the faulty line is opened by the protecting devices. Fig. 6.5 shows the responses for the normal operating point, i.e., $P_o = 0.9$ pu with a power factor of 0.9 lagging and $E_B = 1.0$ pu. The transmission line parameters are $R_e = 0.03$ pu and $X_e = 0.15$ pu, which is a strong connection to the infinite bus. As is seen from the figure, the responses of the fixed-parameters FPSS and the NFPSS are close to each other. The reason is that the FPSS was designed for this operating condition and both stabilisers have nearly optimum responses in this case.

Now, assume that the system configuration and operating condition has changed. The new operating condition is $P_o = 0.7$ pu with a power factor of 0.7 lagging and $E_B = 1.0$ pu. The transmission line parameters are $R_e = 0.05$ pu and

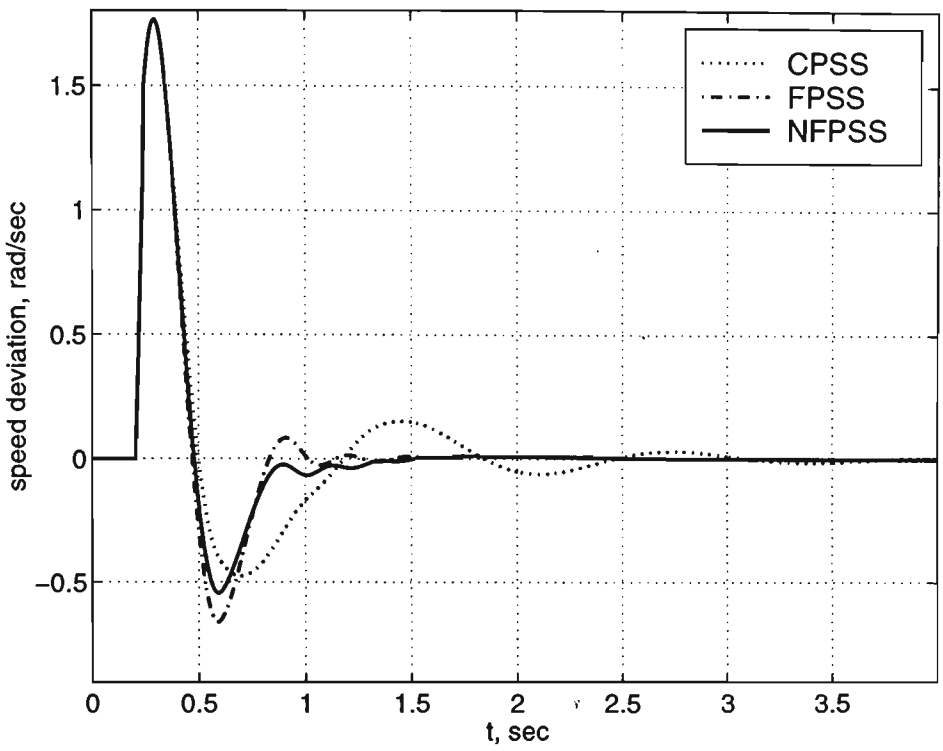


Fig. 6.5 Speed deviation responses of the FPSS and NFPSS for a heavy load with high power factor and strong connection

$X_e = 0.45$ pu, which is a weak connection to the infinite bus. The speed deviation responses of the FPSS and NFPSS for this case are shown in Fig. 6.6. As is seen from the figure, the response of the fixed-parameters FPSS is very oscillatory. It shows the degradation of the FPSS for the new configuration. On the other hand, the response of the system with the NFPSS is optimum in this case, too. This and other results (not shown here) reveal the robustness of the NFPSS for a wide range of operating conditions.

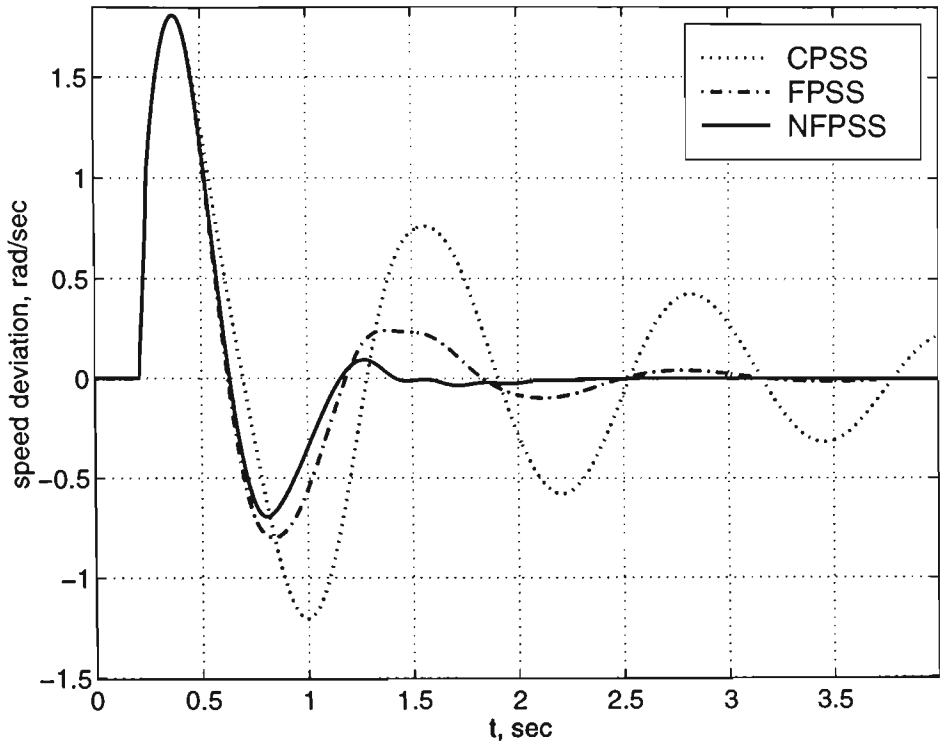


Fig. 6.6 Speed deviation responses of the FPSS and NFPSS for a light load with low power factor and weak connection

6.4 On-Line Tuning of the FPSS using a Fuzzy Logic System

6.4.1 Introduction

A FLS can be used instead of ANNs to tune the scaling factors of the FPSS. The advantage is that there is no need for a complete set of training data. A set of linguistic rules which can be obtained from the experts is enough for this scheme. However, the processing time for getting the optimum scaling factors may be longer compared to the ANN tuner.

6.4.2 Tuning scheme

The FPSS is tuned by computing optimum K_p and K_d, using a FLS as shown in Fig. 6.7. Inputs and outputs of the FLS are the same as the ANN tuner. The

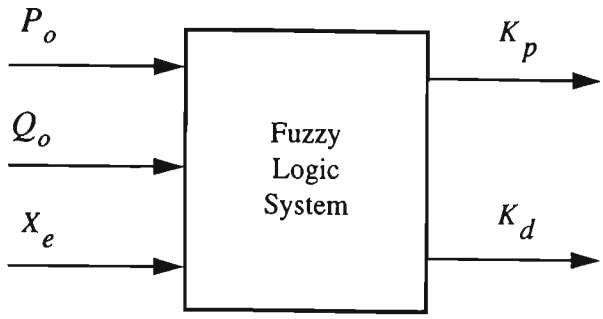


Fig. 6.7 Fuzzy logic system configuration for tuning the FPSS

membership functions for inputs and outputs are shown in figures 6.8-6.12.

From various combinations of the input data as shown in Table 6.1, the optimal parameters of the FPSS are searched. Then the rule base for the FLS is obtained as shown in Table 6.2. For input P_o, three labels are defined, namely, SM, MD, and BG which stand for small, medium, and big, respectively. Similar labels are defined for the input X_e. For input Q_o, three labels are defined: NG, PS, and PB which stand for negative, positive small, and positive big, respectively. For each of the outputs K_p

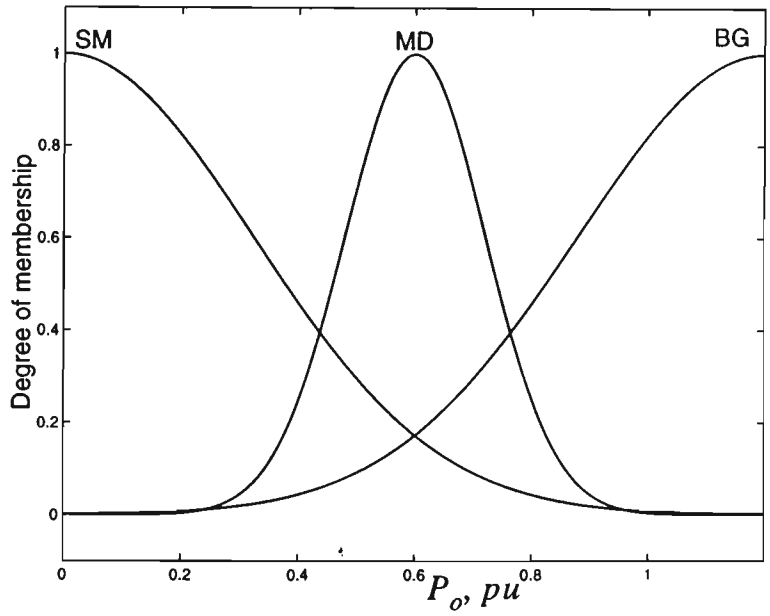


Fig. 6.8 Membership functions for the active power

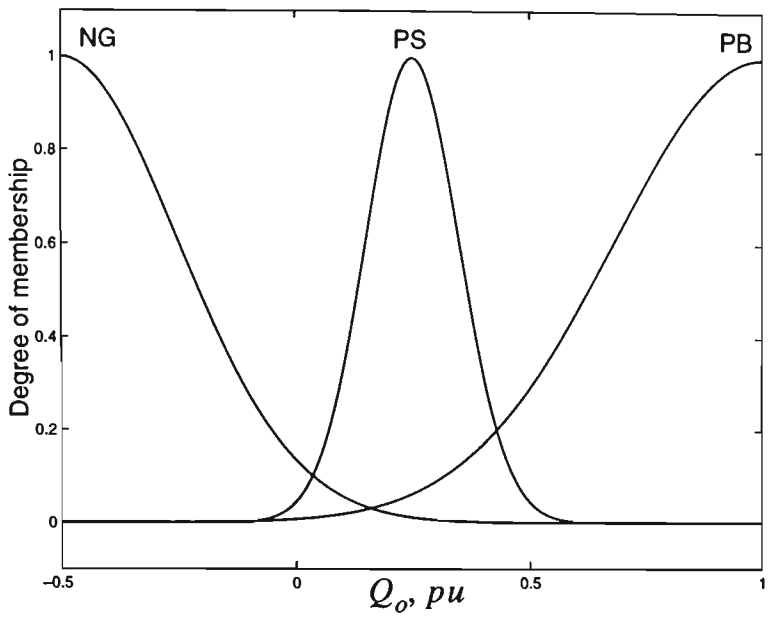


Fig. 6.9 Membership functions for the reactive power

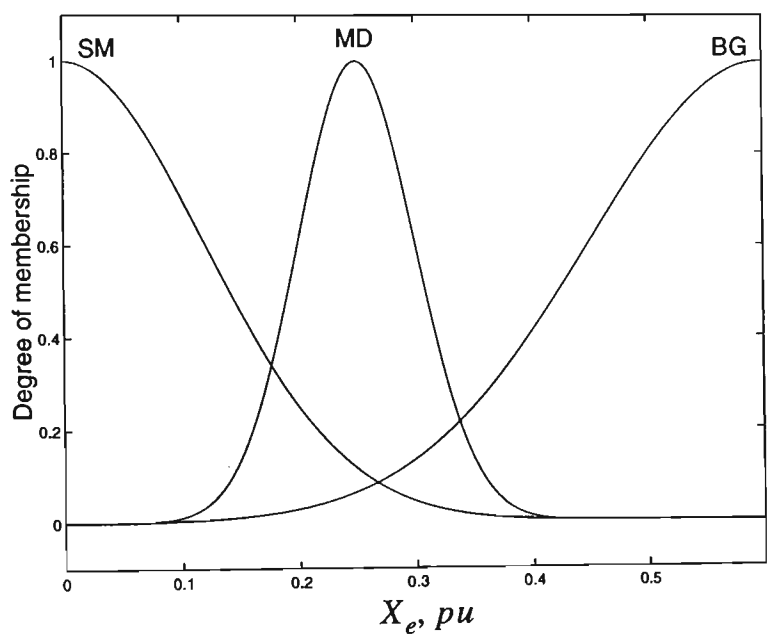


Fig. 6.10 Membership functions for X_e

and K_d, five labels are defined, namely, VS, SM, MD, BG, and VB which stand for very small, small, medium, big, and very big, respectively.

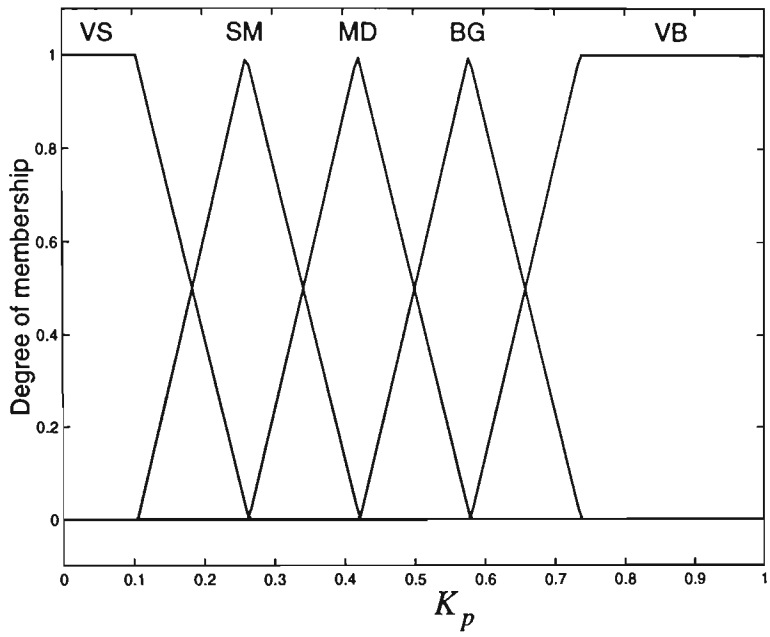


Fig. 6.11 Membership functions for K_p

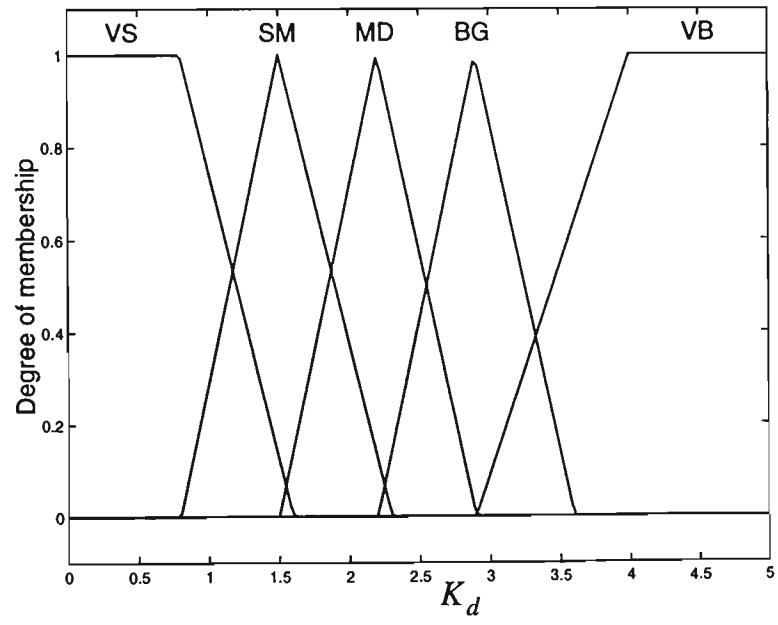


Fig. 6.12 Membership functions for K_d

Note that, although the labels used for different variables look similar in notation, they are actually different. For example, SM for K_p means a number

Table 6.2. Rule base for the FLS with three inputs and two outputs

P_o	Q_o						X_e
	NG		PS		PB		
SM	VS	VS	MD	SM	MD	BG	SM
SM	MD	VS	BG	MD	BG	VB	MD
SM	BG	MD	VB	BG	VB	VB	BG
MD	VS	VS	SM	SM	MD	MD	SM
MD	SM	SM	MD	MD	BG	BG	MD
MD	MD	MD	BG	BG	VB	VB	BG
BG	VS	VS	VS	MD	VS	MD	SM
BG	SM	MD	SM	BG	MD	BG	MD
BG	MD	BG	BG	BG	BG	VB	BG
	K_p	K_d	K_p	K_d	K_p	K_d	

roughly between 0.18 and 0.35 whereas SM for K_d means a number roughly between 1.1 and 1.8. The FLS with three inputs (P_o, Q_o and X_e) and two outputs (K_p and K_d) designed this way is called the *tuner*. It will take the inputs on-line (as the machine is running) and calculate the optimum scaling factors for the FPSS. The whole stabiliser obtained is called *tuned fuzzy power system stabiliser* (TFPSS).

6.4.3 Simulation results

The responses of the fixed-parameters FPSS and TFPSS for the same operating conditions introduced in Section 6.3.4 are obtained here. Fig. 6.13 shows the responses for the normal operating point, i.e., P_o = 0.9 pu with a power factor of 0.9 lagging and E_B = 1.0 pu. The transmission line parameters are R_e = 0.03 pu and

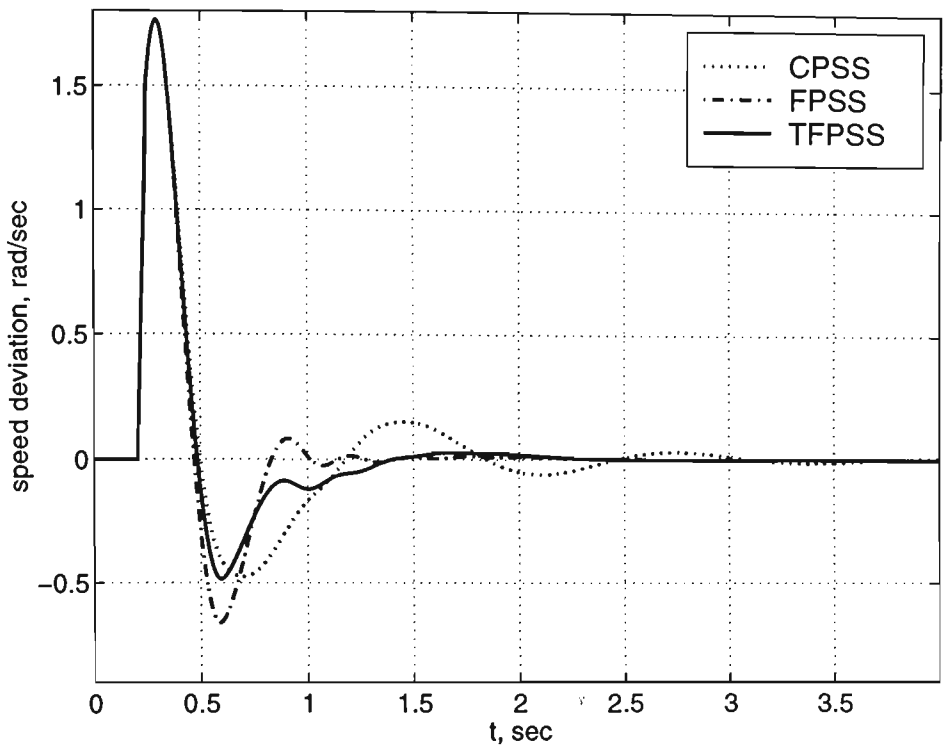


Fig. 6.13 Speed deviation responses of the FPSS and TFPSS for a heavy load with high power factor and strong connection

$X_e = 0.1$ pu.

In the second case, the operating condition is $P_o = 0.7$ pu with a power factor of 0.7 lagging and $E_B = 1.0$ pu. The transmission line parameters are $R_e = 0.05$ pu and $X_e = 0.45$ pu. The speed deviation responses of the FPSS and TFPSS for this case are shown in Fig. 6.14.

By comparing Figures 6.13 and 6.14 with Figures 6.5 and 6.6, it is observed that tuning a FPSS by a FLS the performance of the FPSS is improved. The results are very similar to tuning a FPSS by an ANN. The advantage of tuning by a FLS compared with tuning by an ANN is the fact that there is no need for a complete set of training data. Obtaining such a training set of data for a practical power system may be cumbersome.

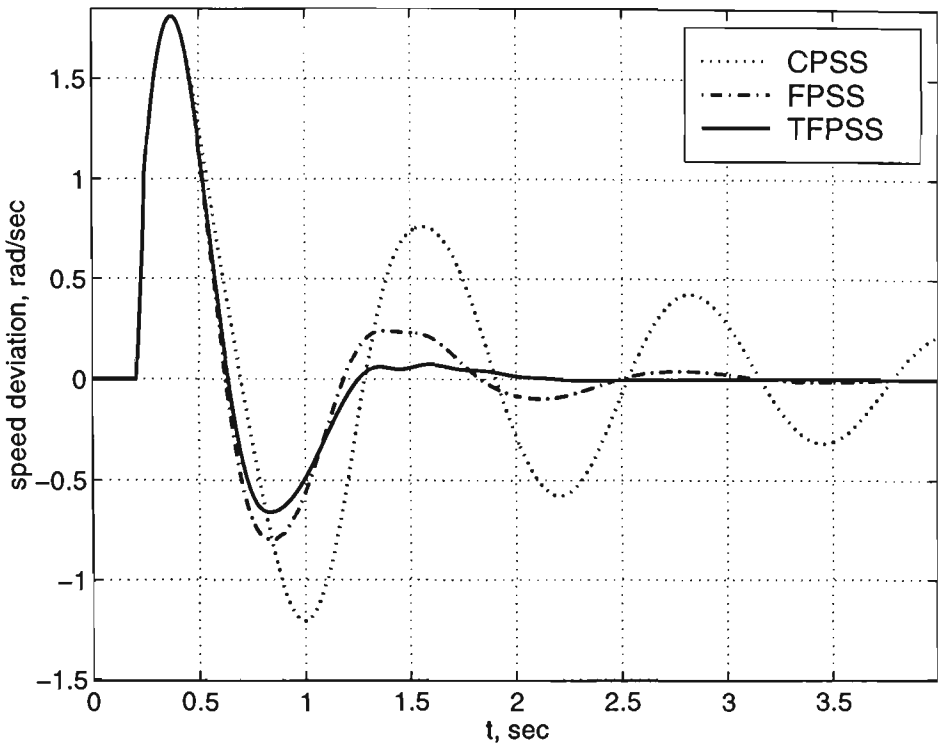


Fig. 6.14 Speed deviation responses of the FPSS and TFPSS for a light load with low power factor and weak connection

6.5 Conclusions

The performance of a fixed-parameter FPSS depends on the operating conditions of the power system, although it is more robust than a CPSS. In particular, when there is a big change in system configuration and operating conditions of the system, the performance of the FPSS will degrade. In order to make the FPSS more robust and adjustable to the new situation, on-line tuning of the FPSS is proposed.

In this chapter, two schemes of on-line tuning of the FPSS was explained. In the first scheme, an ANN was employed as an on-line tuner. A complete set of training data was obtained. This training set gives the optimum parameters of the FPSS for different conditions. Based on this set of training data an ANN was trained off-line with an efficient training algorithm. Then, the ANN was used to tune the

FPSS on-line.

In the second scheme, a fuzzy logic system was used as the tuner. For this case, there is no need for a complete set of training data. Instead, a few linguistic rules about the relationship between the parameters of the FPSS and different operating conditions are used. These rules can be obtained from power system experts.

The advantage of the ANN tuner is that it is faster than the FLS tuner, therefore, it is more suitable for implementation. The disadvantage of the NFPSS is the fact that for obtaining the training data a number of different tests should be performed on the power system. These tests may be cumbersome in practice and may not cover all practical configurations and operating conditions.

Chapter 7

Neural-Network Based Adaptive Power System Stabiliser

7.1 Overview of the Chapter

This chapter explains how a hierarchical structure of artificial neural networks can be used to build an adaptive power system stabiliser.

7.2 Introduction

The tuning schemes introduced in Chapter 6 are basically gain-scheduling techniques to adjust the FPSS to different operating conditions of the power system. In this and the next chapter an attempt has been made to design adaptive power system stabilisers (APSSs) based on artificial neural networks (ANNs) and fuzzy logic. These APSSs will adapt themselves to the new operating conditions based on the input-output response of the system.

This chapter presents an adaptive power system stabiliser based on multi-layer feedforward ANNs. A hierarchical architecture of ANNs, consisting of two sub-networks, is used. One sub-network is used for the identification of a nonlinear

power plant and the other one is used as a stabiliser. The weights of the ANN stabiliser are adjusted according to the difference between the output of the ANN identifier and a desired output track (which can be the output of a reference model). The NNS and neural network identifier are trained in different stages by the backpropagation algorithm. An application of this scheme for regulating the speed of a synchronous power generator under disturbance and fault conditions is described.

The nonlinear mapping properties of ANNs make them a good candidate for the identification and control of nonlinear systems. The theoretical results obtained by Cybenko [186] and Funahashi [187] have proven that a wide range of nonlinear functions can be closely approximated by a feedforward neural network (FNN) with only a single hidden layer of nonlinear elements. Various schemes, differing in the way that ANNs are employed in the implementation of the identification and control algorithms for realising the adaptive control objectives, have been incorporated. For control problems where reference models are used to specify desired dynamical response, novel structures constructed from multi-layer ANNs have been developed by Narendra and Parthasarathy [188, 189]. Observability, identification and control of nonlinear dynamical systems using FNNs have also been discussed by Levin and Narendra [190]. A different approach that utilises self-tuning mechanisms for control adaptation is proposed by Chen [191] using FNNs.

Literature survey reveals that there are applications of ANNs as adaptive PSSs [89, 90]. In reference [90] the ANN is trained off-line and imitates the behaviour of a self-optimising pole shifting adaptive PSS. Thus *a priori* knowledge of the plant is required and basically the design is based on linear schemes. The adaptive neural network power system stabiliser (ANNPSS) proposed in this paper is similar to the work done by Wu et.al. as described in reference [89]. The difference being that Wu et.al. have designed an ANN regulator which affects both the excitation system and

the governor of a generator, while the proposed scheme affects only the extra damping signal in the excitation system. Since the governor is slow in action compared with the PSS, the speed of the generator is mainly affected by the PSS in the short period after the application of a major disturbance, such as a fault occurring in the transmission lines.

7.3 Adaptive power system stabiliser

7.3.1 General adaptive control

A general adaptive control structure is shown in Fig. 7.1. There are two loops present in this structure, the control loop and the parameter adjustment loop. The latter attempts to adjust the control gains to force the plant to follow some desired response. Common strategies in adaptive control are usually based on the assumption that the plant can be linearised about a given operating point to produce an

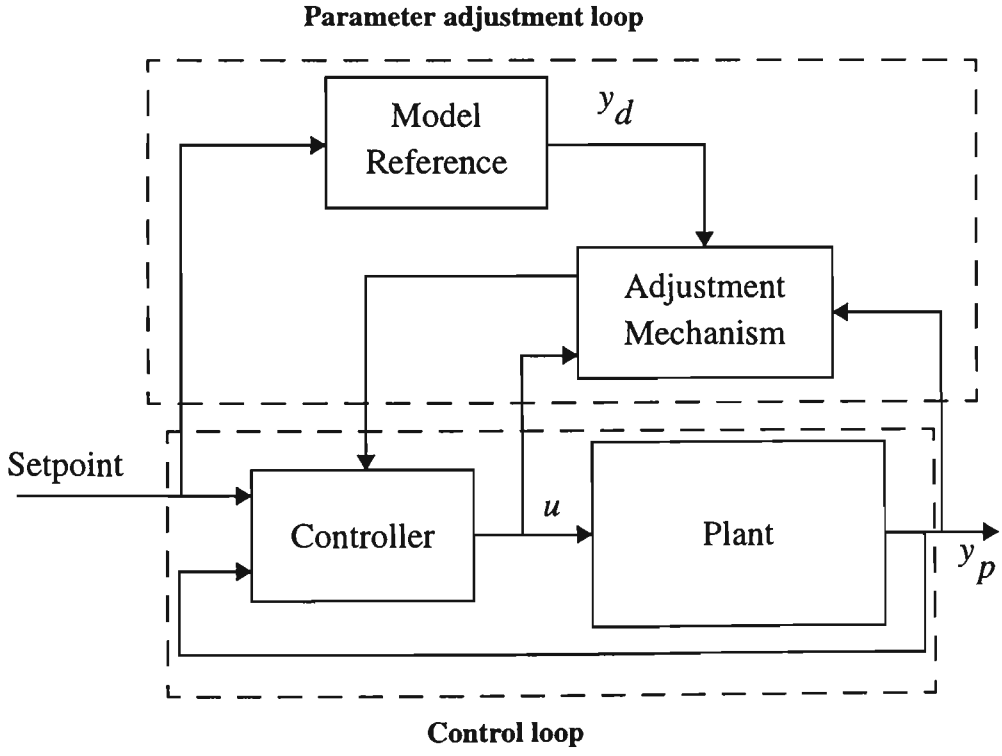


Fig. 7.1 General model reference adaptive control structure

approximate linear model. However, in practice, industrial systems are highly nonlinear and have unmodelled dynamics. Hence nonlinear modelling techniques and control are more suitable for controlling such systems.

7.3.2 Neural networks for adaptive control

FNNs consist of simple processing elements arranged in layers. Each element takes as input the weighted sum of all the outputs of the previous layer and passes this through a nonlinear activation function. Training the network involves adjusting the weights, using some learning rule, so that the network emulates the desired nonlinear mapping from the input to the output vector.

The nonlinear mapping properties of ANNs are central to their use in control engineering. FNNs can be readily thought of as performing an adaptive, nonlinear vector mapping. The FNN can therefore be utilised as a general structure for an adaptive nonlinear controller. A possible control scheme based upon such an ANN controller is shown in Fig. 7.2.

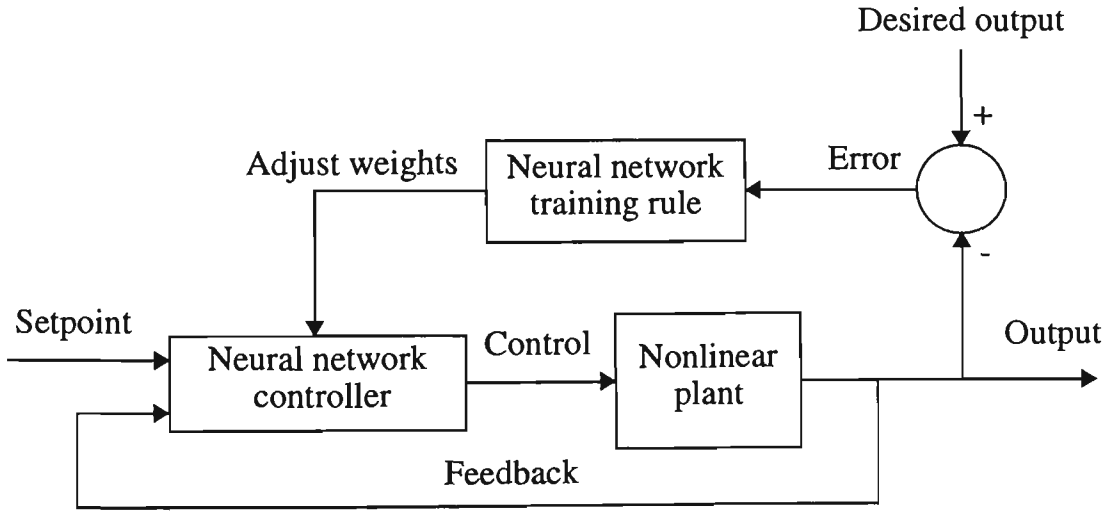


Fig. 7.2 Neural network adaptive control scheme

7.4 Training Techniques

For the formulation of adaptive control problem as shown in Fig. 7.2, it is necessary to adjust the weights of the ANN. During the learning phase, the weights are adjusted in such a manner as to minimise some cost function of the plant output and the desired response.

As shown in Fig. 7.2, the plant is situated between the neural network controller and the error. It is necessary to find some method by which the error at the output of the plant can be fed back, to produce a suitable descent direction at the output of the ANN.

Nguyen and Widrow [192] and Jordan and Jacobs [193] proposed using a FNN model of the plant as a channel for the backpropagation of errors to the neural controller. This concept will be illustrated in Section 7.6. An ANN is first trained to provide a model of the nonlinear plant. This can then be used in parallel with the plant, with errors at the plant output backpropagated through the model to form the necessary gradients at the output of the neural network controller.

7.5 Dynamic Modelling Using Neural Networks

The nonlinear plant, i.e. the synchronous machine, can be represented by the discrete-time, nonlinear, ANN model as shown below:

$$\Delta\hat{\omega}(k+1) = \hat{F}[y(k), \ldots, y(k-p), e_{fd}(k), \ldots, e_{fd}(k-q)] \tag{7.1}$$

where $\{y(k)\}$ is the output vector, $\{e_{fd}(k)\}$ is the input vector and p, q are the orders of the time series $\{y(k)\}$ and $\{e_{fd}(k)\}$. $\Delta\hat{\omega}$ is the output of the neural network identifier (NNI) which predicts the plant output. The plant can then be

dynamically modelled by adjusting the network weighting matrix $W_m(k)$, such that the following cost function is minimised:

$$J_m(k) = \sum_{i=k-N_m+1}^{k} [\Delta\omega(i) - \Delta\hat{\omega}(i)]^2 \tag{7.2}$$

where $\Delta\omega$ is the plant output and $\Delta\hat{\omega}$ is the ANN output. The number N_m is defined as $N_m = \frac{T_m}{T_s}$, where T_m is the period over which the mapping performance is to be evaluated and T_s is the sampling period. Backpropagation is used to provide the necessary gradient vector of the cost function with respect to the weights and the network weighting matrix is then updated as follows:

$$W_m(k) = W_m(k-1) - \frac{\lambda_m}{N_m} \sum_{i=k-N_m+1}^{k} \nabla J_m(W_m, i) \tag{7.3}$$

where $\nabla J_m(W_m, i)$ is the gradient vector at the ith sampling and λ_m is the learning rate. Inherent in equation (7.3) is the averaging of the gradient vector over the period T_m.

The sampling period was chosen to be 20 msec and the number N_m was selected to be 10 samples. The synchronous machine was initialised with the real and reactive powers being $P_o = 0.8$ pu and $Q_o = 0.3$ pu. A pseudo-random signal was then employed to excite the NNI. The response shown in Fig. 7.3 shows the performance of the neural dynamic modeller after training is completed.

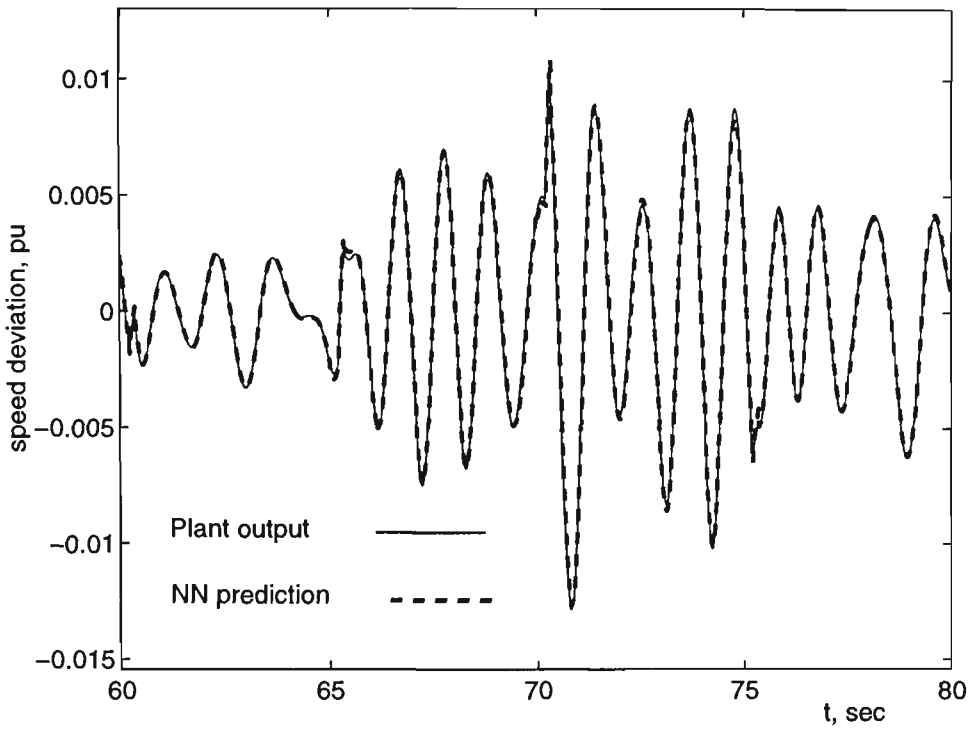


Fig. 7.3 Speed deviation prediction

7.6 Adaptive Neural Network PSS

The PSS required to stabilise the synchronous machine may be defined as:

$$u(k) = G[\Delta\omega(k-1), \ldots, \Delta\omega(k-s), \Delta P(k-1), \ldots, \Delta P(k-r)] \tag{7.4}$$

The task is then to choose an ANN *G(.)* which will minimise the following cost function:

$$J_c = \sum_{i=k-N_c+1}^{k} [d(i) - \Delta\hat{\omega}(i)]^2 \tag{7.5}$$

where $d(i)$ is the ith sample of the desired output track and $\Delta\hat{\omega}(i)$ is the ith sample of

the predicted speed deviation by the NNI. $N_c = \frac{T_c}{T_s}$, where T_c is the period over which the neural network should be adjusted and T_s is the sampling period. Here T_c was chosen to be equal to two sampling periods, such that N_c is equal to two. N_c should be less than N_m, which means that the NNS will be adjusted more frequently than the NNI. By choosing $N_m = 10$ and $N_c = 2$, the NNI will be adjusted after 5 successive adjustments of the NNS.

The ANNPSS block diagram is shown in Fig. 7.4. This hybrid ANNPSS consists of two sub-networks, a neural network identifier (NNI) and a neural network stabiliser (NNS). In equation (7.1) p is chosen to be equal to two and q to be equal to one. In equation (7.4) r and s are both chosen to be equal to one. Choosing small values for p, q, r and s eases the computation requirements. This in turn makes the implementation feasible. Bigger numbers would make the identification more accurate, however, it would complicate the implementation.

The NNI sub-network acts as the dynamic model of the plant, providing a

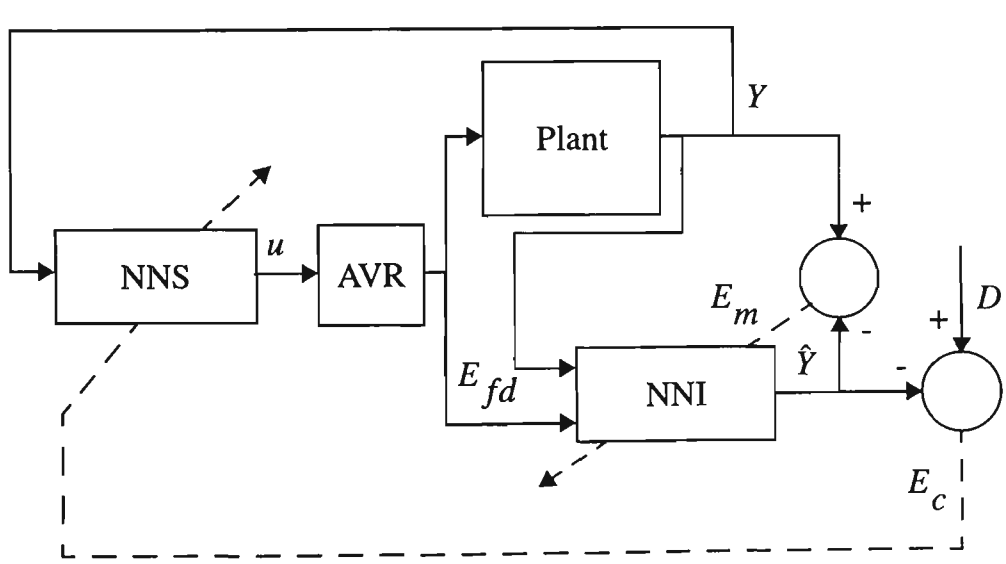


Fig. 7.4 Adaptive neural network PSS

channel through which the errors could be backpropagated to adjust the weights of the NNS. In order to compute the gradient vector $\nabla J_c(W_c, k)$ one needs to compute $\frac{\partial J_c}{\partial \theta_j}$, where θ_j is a typical parameter (a weight) of the NNS. In order to do that, $\frac{\partial E_c}{\partial \theta_j}$ should be computed. E_c is the difference (error) between the desired output (D) and the output of NNI ($\hat{Y}$). $\frac{\partial E_c}{\partial \theta_j}$ can be computed as follows:

$$\frac{\partial E_c}{\partial \theta_j} = \frac{\partial E_c}{\partial u} \cdot \frac{\partial u}{\partial \theta_j} \tag{7.6}$$

$\frac{\partial u}{\partial \theta_j}$ can be computed using backpropagation through NNS. Since AVR is a linear transfer function one can show that

$$\frac{\partial E_c}{\partial u} = \frac{\partial E_c}{\partial E_{fd}} \cdot \frac{\partial E_{fd}}{\partial u} = \frac{\partial E_c}{\partial E_{fd}} \cdot AVR(s) \tag{7.7}$$

Since $\frac{\partial E_c}{\partial E_{fd}}$ can be computed at any instant using backpropagation through NNI, $\frac{\partial E_c}{\partial u}$ can be realised as the output of AVR whose input is $\frac{\partial E_c}{\partial E_{fd}}$.

The NNS network weighting matrix is then updated as

$$W_c(k) = W_c(k-1) - \frac{\lambda_c}{N_c} \sum_{i=k-N_c+1}^{k} \nabla J_c(W_c, i) \tag{7.8}$$

The hierarchical structure for the ANNPSS is illustrated in Fig. 7.5. For the NNS a (2,10,1) ANN was implemented. The notation (2,10,1) means that the ANN consists of two input nodes, ten hidden neurons and one output node. Also a (7,20,1) ANN was employed as the NNI. The sigmoid activation function, i.e. $f(x) = \frac{1-e^{-x}}{1+e^{-x}}$ was used for the hidden layers of both neural networks. More hidden nodes would make the ANNs more efficient, but this would cause more complexity which means more costly implementation. Twenty hidden neurons were chosen for the NNI compared with ten hidden neurons for the NNS, because The plant is more complicated than the stabiliser.

The weights in the NNS were updated such that the error between the output of

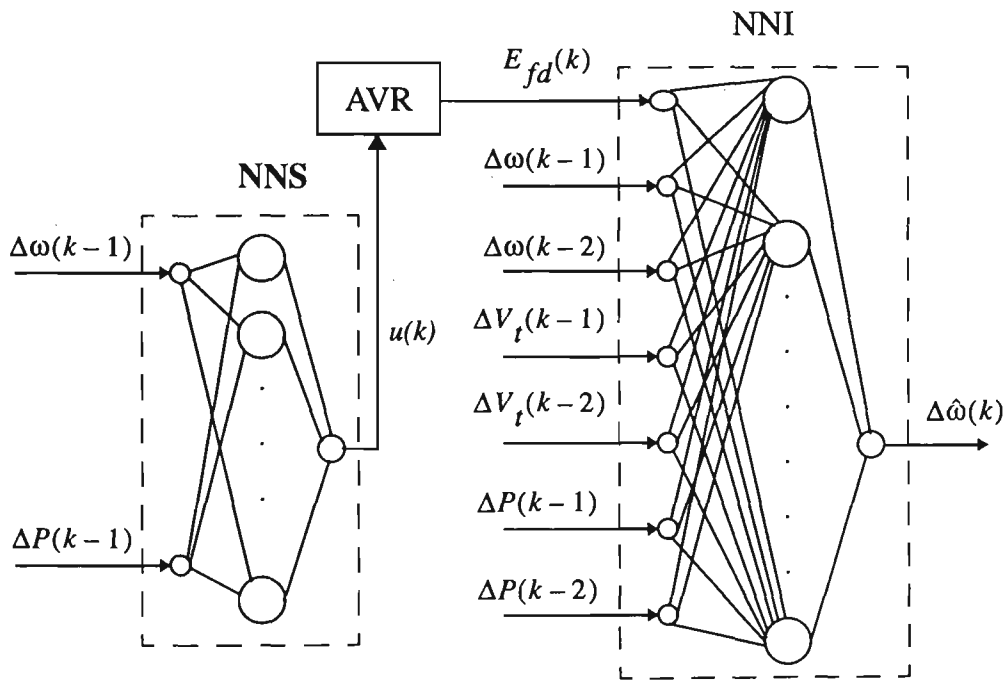


Fig. 7.5 Hierarchical architecture for the NN PSS

the NNI and the desired track is minimised. The desired track is the output of a predictor which is designed such that it eventually reduces the speed deviation of the machine to zero.

7.6.1 Design of the predictor

A predictor is designed on the basis of guiding the disturbed output variable of the synchronous machine step by step to a desired steady equilibrium point. In other words, a desired trace of the output from t_i to t_{i+1} can be predicted, based on the present and past-time values of the states of the machine. Generally, the equation for a predictor is of the form shown below:

$$X(k+1) = A_0X(k) + \ldots + A_NX(k-N) \tag{7.9}$$

where $X(k)$ is the state of the system at time k. $A_i(i = 0, 1, \ldots, N)$ are chosen so that any disturbed output variable always transfers towards the desired steady equilibrium point. It should be pointed out that $A_i(i = 0, 1, \ldots, N)$ describe the relationship between the desired outputs of the predictor and the outputs of the dynamic system, and may be chosen according to the qualitative requirements of the controlled system.

In the case under study the disturbed output is $\Delta\omega$. The next sample of $\Delta\omega$ is predicted using the present samples of $\Delta\omega$ and ΔP, which are the measured quantities of the plant. So a predictor in this case is easily designed as follows:

$$\Delta\tilde{\omega}(k+1) = \alpha \cdot \Delta\omega(k) + \beta \cdot \Delta P(k) \tag{7.10}$$

In this equation $\Delta\tilde{\omega}$ is the output of the predictor. α and β are chosen such that the system described by equation (7.10) is globally asymptotically stable.

7.7 Simulation Results

The ANNPSS has the capability of learning while the machine is operating. In order to show this capability the system response to a three-phase to ground fault is simulated with a random initialisation of the NNS. The NNS will learn how to adjust itself to damp the speed deviation oscillations during the post-fault period. In simulation studies the fault will be applied again to the system to show the improved response of the ANNPSS. In practice, the NNS can be trained before the real-time field application.

The computer simulations are performed for the normal operating point, i.e., P_o = 0.9 pu with a power factor of 0.9 lagging and E_B = 1.0 pu. The transmission line parameters are R_e = 0.03 pu and X_e = 0.15 pu. Fig. 7.6 shows the speed deviation response of the ANNPSS to a three-phase to ground fault with a duration of 40 msec when the NNS is initialised randomly. To show that the ANNPSS has learned how to damp the oscillations more clearly, the output of the ANNPSS is shown in Fig. 7.7. At first the stabiliser output is close to zero, because the NNS weighting matrix is initialised with very small random numbers. Then the ANNPSS learns how to stabilise the system during the post-fault period.

Now, the fault is applied again after the learning phase. The system response is shown in Fig. 7.8. Obviously, the performance of the ANNPSS has improved. The output of the ANNPSS after the learning phase is shown in Fig. 7.9.

Clearly the learning capability of ANNs makes the design of the ANNPSS possible without knowing much about the system dynamics and machine parameters.

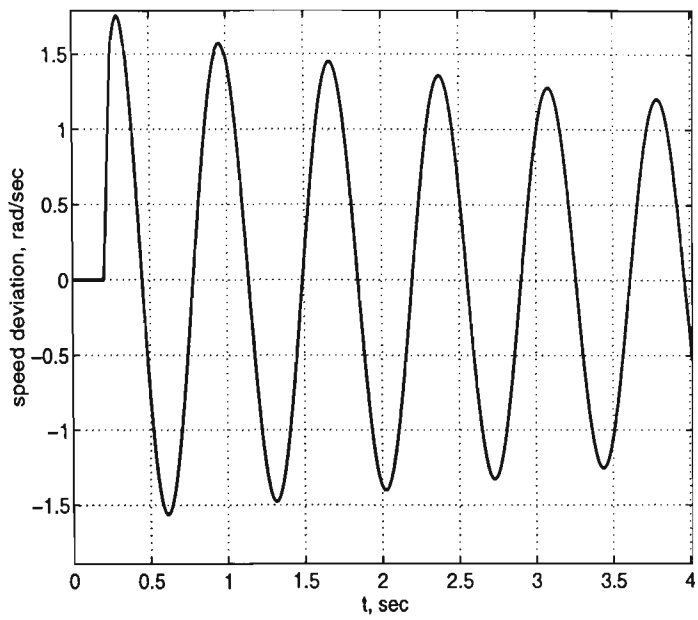


Fig. 7.6 Speed deviation response of the system with the ANNPSS by random initialization of the NNS

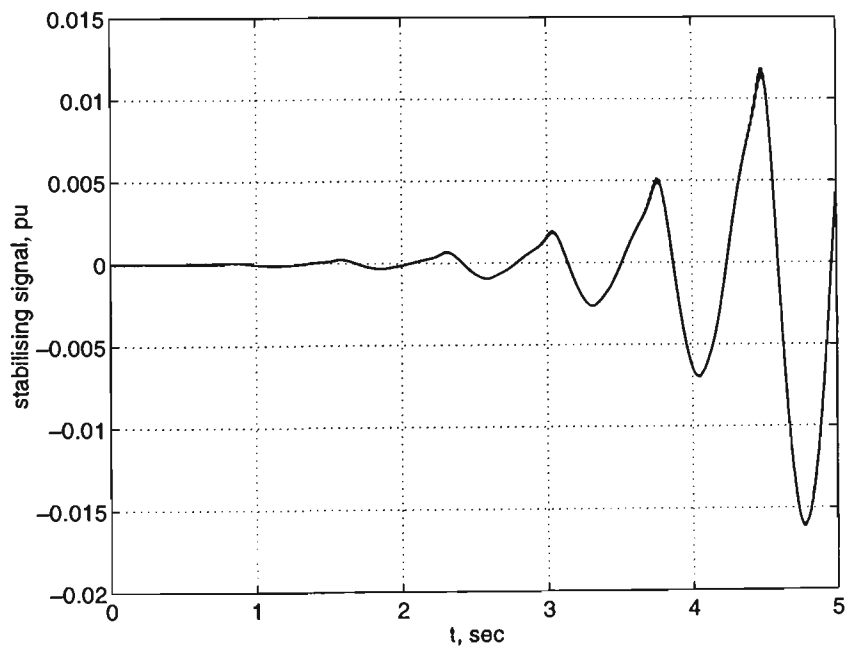


Fig. 7.7 The output of the ANNPSS during the learning phase

All the adjustments of NNI and NNS are done based on measurements of inputs and outputs of the system while it is operating.

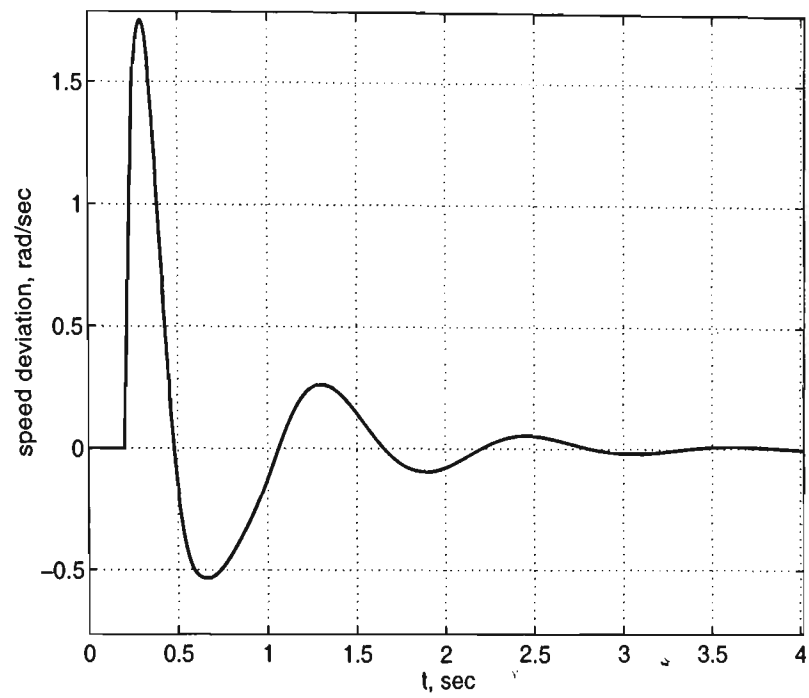


Fig. 7.8 Speed deviation response of the system with the ANNPSS after the learning phase

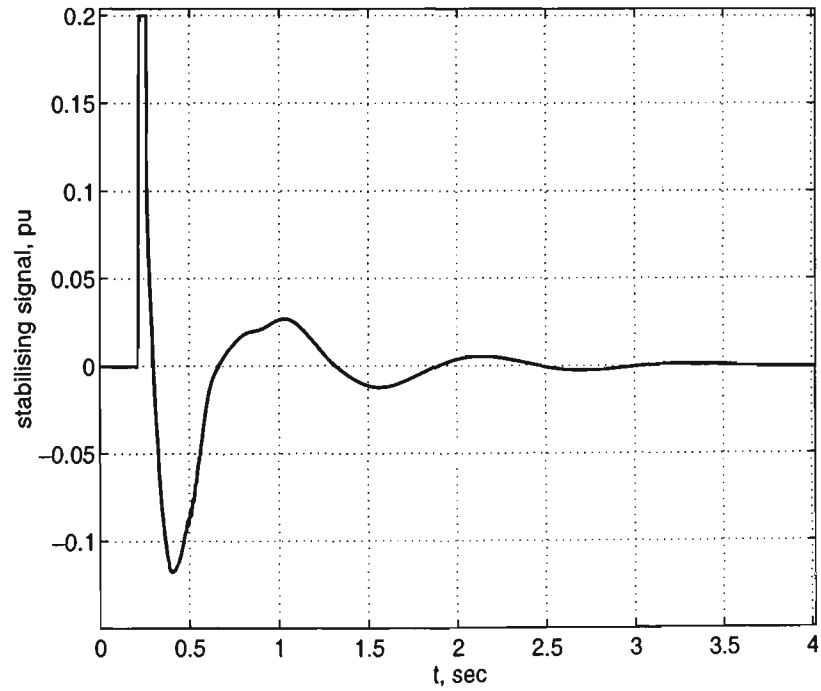


Fig. 7.9 The output of the ANNPSS after the learning phase

7.8 Conclusions

This chapter has presented a hierarchical structure of ANNs as an adaptive power system stabiliser. This consists of two sub-networks for identification and adaptive control. The identification and adaptive control performance have been evaluated by simulation studies on a nonlinear model of a synchronous machine. One of the simulation results was shown to illustrate the learning capability of ANNPSS. It is shown that the ANNPSS is capable of learning how to damp the oscillations in the speed response of the machine, while the system is operating.

The learning capability of ANNs is a great advantage. The design of the ANNPSS is not based on a complete (mathematical) knowledge of system dynamics and machine parameters. It will be adjusted using the input-output relationships of the machine, while it is operating.

In spite of the above-mentioned advantage of the ANNPSS, it has a big disadvantage which should be mentioned here. It is possible that while the ANNs are learning during a major disturbance, the system goes unstable. In some situations the output of the NNI may be very close to the output of the plant, but the relationship between E_{fd} and $\Delta\hat{\omega}$ may not be the same as the relationship between E_{fd} and $\Delta\omega$. This may cause the NNS to adjust its weights in a wrong direction, which may force the system to instability.

Another major disadvantage of the ANNPSS is that a lot of practical tests are required to be performed on the power system for training purposes. These tests may not be feasible in some situations. However, these tests may be feasible for the cogenerator plants, which this study is all about.

This major disadvantages of the ANNPSS make it a poor alternative to other PSSs as an independent power system stabiliser. It may be possible to use the

ANNPSS in combination with other PSSs to improve the system performance. However, this leads to a highly complicated stabiliser, which is undesirable.

Chapter 8

Adaptive Fuzzy Logic Power System Stabilisers

8.1 Overview of the Chapter

The major concern about the ANNPSS as explained in Chapter 7 was the possibility of system instability during learning phase of the ANNs. In this chapter a technique is sought to design adaptive power system stabilisers which can guarantee system stability. It seems that the adaptive fuzzy logic controllers introduced recently by Wang [95, 194] are very suitable for this purpose.

In this chapter, two adaptive fuzzy logic power system stabilisers (AFPSSs) are developed using the concept of fuzzy basis functions. In the first scheme, which is known as an indirect adaptive scheme, power system is modelled using differential equations with parameters which are nonlinear functions of the state of the system. These nonlinear functions may not be known, however, some linguistic information is available about them. Utilising this information, fuzzy logic systems (FLSs) are designed to model the system behaviour. The control law is obtained using the uncertainty principle. Based on the Lyapunov's synthesis method, adaptation rules are developed to make the controller adaptive to changes in operating conditions of

the power system. The simulation studies are carried out for an industrial cogenerator and utilise a one machine-infinite bus model.

In the second scheme, a direct adaptive fuzzy logic power system stabiliser is developed. The linguistic rules, regarding the dependence of the plant output on the controlling signal, are used to build the initial FPSS. This is different with the first scheme because the linguistic rules are directly used to construct the AFPSS. Based on the Lyapunov's direct method, an adaptation rule is developed in order to make the FPSS adaptive to changes in operating conditions of the power system.

8.2 Introduction

Wang and Mendel [166] introduced fuzzy basis functions (FBFs) which have the capability of combining both numerical data and linguistic information. These FBFs are quite general. Their exact mathematical structure depends on four choices that one must take to design any fuzzy logic system, namely, type of fuzzification, membership function, inference mechanism, and defuzzification strategy.

Further, Wang used the concept of FBFs to introduce stable adaptive fuzzy logic control of nonlinear systems [195]. His initial adaptive fuzzy logic controller is constructed from the fuzzy IF-THEN rules provided by human experts and some arbitrary rules. An adaptive law is then used to update the parameters of the adaptive fuzzy logic controller during the adaptation procedure. If the fuzzy IF-THEN rules from human experts provide good control strategies, then the adaptation procedure will converge very quickly. On the other hand, if there are no linguistic rules from human experts, then his adaptive fuzzy logic controller becomes a regular nonlinear adaptive controller, similar to the radial basis function adaptive controller [196]. He applied the adaptive fuzzy logic controller to an unstable first-order nonlinear system and a second-order chaotic system [195]. He also applied a stable indirect adaptive

fuzzy logic controller to an inverted pendulum tracking problem [197].

In Section 8.4, an indirect adaptive fuzzy logic power system stabiliser (AFPSS) is designed. Firstly two fuzzy logic systems (FLSs) are designed which are the approximations for the unknown nonlinear functions present in the model of the power system. Then these FLSs are used to obtain the control law whose role is to stabilise the power system. The FLSs are adapted based on the Lyapunov's synthesis method. The adaptation law is such that the difference between the output of the plant and a desired track will be minimised.

In Section 8.5 a direct adaptive fuzzy logic power system stabilizer (AFPSS) is designed. The direct AFPSS is different from the indirect AFPSS in the sense that the linguistic rules describe the behaviour of the PSS directly.

The AFPSSs are applied to a cogenerator system. The nonlinear simulations for a one machine-infinite bus system reveals that the adaptation procedure prevents large degradation of system performance for a wide range of operating conditions. This makes the AFPSS superior to a CPSS or a fixed-parameter FPSS.

8.3 Adaptive Fuzzy Logic Control

The basic configuration of an adaptive fuzzy logic control system is shown in Fig. 8.1 [194]. The reference model is used to specify the ideal response that the fuzzy logic control system should follow. The plant is assumed to have unknown components. The fuzzy logic controller is constructed from fuzzy logic systems whose parameter vector $\underline{\theta}$ is adjustable. The adaptation law adjusts the parameter vector on line such that the plant output $y(t)$ tracks the reference model output $y_m(t)$.

Wang [194] has made a comparison between adaptive fuzzy logic control,

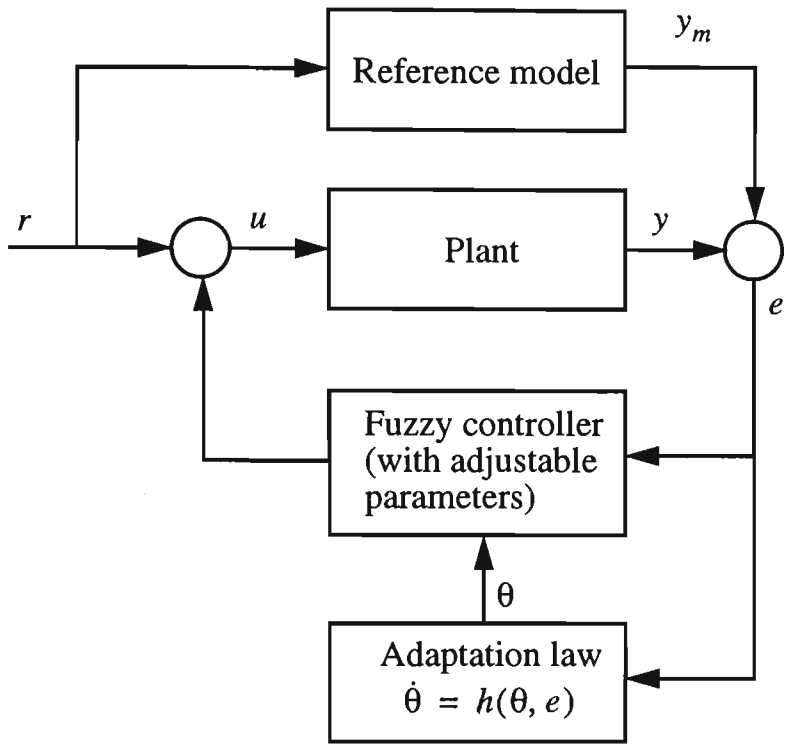


Fig. 8.1 The basic configuration of adaptive fuzzy control systems

conventional adaptive control and non-adaptive fuzzy logic control systems. These comparisons are summarised in Table 8.1. In this table adaptive fuzzy logic controller is referred to as AFLC, conventional adaptive controller is referred to as CAC, and non-adaptive fuzzy logic controller is referred to as FLC.

The main advantages and disadvantages of adaptive fuzzy logic control over non-adaptive fuzzy logic control are summarised in Table 8.2.

8.3.1 Direct and indirect adaptive fuzzy logic control

In the adaptive control literature, adaptive controllers fall into two categories [188]: direct and indirect. In direct adaptive control, the parameters of the controller are directly adjusted to reduce some norm of the output error between the plant and the reference model. In indirect adaptive control, the parameters of the plant are estimated and the controller is chosen assuming that the estimated parameters

Table 8.1. Similarities and differences between adaptive fuzzy logic control and other schemes

Similarities and differences between:	Similarities	Differences
adaptive fuzzy logic control and conventional adaptive control	(i) the basic configuration and principles are more or less the same. (ii) the mathematical tools used in the analysis and design are very similar	(i) the AFLC is universal for different plants, whereas the structure of a CAC changes from plant to plant. (ii) human knowledge about the plant dynamics and control strategies, stated as some linguistic rules, can be incorporated into AFLC.
adaptive fuzzy logic control and non-adaptive fuzzy logic control	Both use linguistic IF-THEN rules from human experts.	(i) the AFLC changes during real time operation, whereas the FLC is fixed. (ii) an additional component, the adaptation law, is introduced into the AFLC system to adjust its parameters.

represent the true values of the plant parameters.

In fuzzy logic control, linguistic information from human experts can be placed into two categories:

Table 8.2. Advantages and disadvantages of adaptive fuzzy logic control

Advantages and disadvantages of:	Advantages	Disadvantages
adaptive fuzzy logic control over non-adaptive fuzzy logic control	(i) better performance is usually achieved because the adaptive fuzzy logic controller can adjust itself to the changing environment. (ii) less information about the plant is required because the adaptation law can help to learn the dynamics of the plant during real time operation.	(i) The resulting control system is more difficult to analyse because it is not only nonlinear but also time varying. (ii) implementation is more costly.

- fuzzy logic control rules which set forth the situations in which certain control actions should be taken (e.g., if the speed-deviation is negative medium and the acceleration is negative medium then the PSS output should be negative big).

- fuzzy logic IF-THEN rules, which describe the behaviour of the unknown plant (e.g., if the control signal applied to the synchronous machine is negative, then the acceleration of the shaft will change in a positive direction).

Adaptive fuzzy logic controllers which make use of these two classes of linguistic information correspond to the direct and indirect adaptive control schemes, respectively. More specifically, in the direct scheme, linguistic fuzzy logic control rules can be directly used to implement the adaptive fuzzy logic controller. Whereas,

in the indirect scheme, linguistic rules are used to model the plant. The controller is then constructed assuming that the fuzzy logic system approximately represents the plant.

8.4 Indirect Adaptive Fuzzy Logic PSS

In this section the procedure of designing an indirect AFPSS for a cogenerator system is explained. The electrical part of a cogenerator is mainly a synchronous machine. In order to represent the synchronous machine mathematically, let $x_1 = \Delta\omega = speed\ deviation$ and $x_2 = \Delta P = P_m - P_e = accelerating\ power$. It is possible to represent the system with the following nonlinear equations:

$$\begin{aligned} \dot{x}_1 &= a x_2 \\ a\dot{x}_2 &= f(x_1, x_2) + g(x_1, x_2)u \\ y &= x_1 \end{aligned} \tag{8.1}$$

where $a = \frac{1}{2H}$ and H is the per unit inertia constant of the machine. $\underline{x} = [x_1, x_2]^T \in R^2$ is the state vector of the system and can be measured. $f(.)$ and $g(.)$ are unknown nonlinear functions. u is the controlling signal which is the output of the PSS to be designed. Equation (8.1) represents the machine during a transient period after a major disturbance has occurred in the system. It is possible to find two nonlinear functions $f(.)$ and $g(.)$ such that

$$\dot{P}_e = -2H[f(x_1, x_2) + g(x_1, x_2)u] \tag{8.2}$$

Simulation studies show that a positive u will cause a positive change in P_e, i.e., $\dot{P}_e > 0$ whenever $u > 0$. This means that g is a negative function, i.e.,

$$g(x_1, x_2) < 0, \qquad \text{for all } x_1, x_2 \tag{8.3}$$

Equations (8.1) can be equivalently represented as

$$\begin{aligned} \ddot{x} &= f(x, \dot{x}/a) + g(x, \dot{x}/a)u \\ y &= x \end{aligned} \tag{8.4}$$

where $\underline{x} = [x, \dot{x}/a]^T = [x_1, x_2]^T \in R^2$ is the state vector of the system that can be measured.

In the nonlinear control literature [198, 199], the system represented by equation (8.1) or (8.4) is known as in *normal form* and has a *relative degree* equal to two. The control objective is to force y to follow a given desired signal y_m, under the constraint that all signals involved must be bounded. More specifically, the control objective is [195] to determine a feedback control (based on fuzzy logic systems) and an adaptive law for adjusting the parameter vectors such that:

1. The closed-loop system be globally stable in the sense that all variables, must be bounded.

2. The tracking error, $e = y - y_m$, should be as small as possible under the constraints in the previous objective.

In the rest of this section, the procedure to construct an indirect adaptive fuzzy

controller to achieve the above control objectives is discussed. This approach was first introduced by Wang [95]. The same approach is used here with some modifications to suit the PSS design.

To begin, let $\underline{e} = [e, \dot{e}]^T$ and $\underline{k} = [k_2, k_1]^T \in R^2$. Suppose that the two roots of the polynomial $h(s) = s^2 + k_1 s + k_2$ are in the open left half-plane. It is known in control system design theory that if the functions f and g are known, then the control law

$$u^* = \frac{1}{g(\underline{x})}[-f(\underline{x}) + \ddot{y}_m - \underline{k}^T \underline{e}] \tag{8.5}$$

applied to equation (8.4), using u^* instead of u, results in

$$\ddot{e} + k_1 \dot{e} + k_2 e = 0 \tag{8.6}$$

which implies that $\lim_{t \to \infty} e(t) = 0$. This is the main objective of control.

However, f and g are not known. In this section, the objective is to design two FLSs close enough to $f(\underline{x})$ and $g(\underline{x})$ and use them in equation (8.5). Therefore, the control law will be:

$$u_c = \frac{1}{\hat{g}(\underline{x};\underline{\theta}_g)}[-\hat{f}(\underline{x};\underline{\theta}_f) + \ddot{y}_m - \underline{k}^T \underline{e}] \tag{8.7}$$

where $\hat{g}(\underline{x};\underline{\theta}_g)$ and $\hat{f}(\underline{x};\underline{\theta}_f)$ are the FLSs with parameter vectors $\underline{\theta}_g$ and $\underline{\theta}_f$,

respectively.

In order to obtain adaptation laws such that $\hat{f}(\underline{x};\underline{\theta}_f)$ and $\hat{g}(\underline{x};\underline{\theta}_g)$ converge to $f(\underline{x})$ and $g(\underline{x})$ the following procedure is performed:

By adding $\hat{g}(\underline{x};\underline{\theta}_g)u_c$ to both sides of equation (8.1) and using equation (8.7), the following equation is derived:

$$\ddot{e} = -\underline{k}^T \underline{e} + [f(\underline{x}) - \hat{f}(\underline{x};\underline{\theta}_f)] + [g(\underline{x}) - \hat{g}(\underline{x};\underline{\theta}_g)]u_c \tag{8.8}$$

Since $\dot{\underline{e}} = [\dot{e}, \ddot{e}]^T$ equation (8.8) can be written as

$$\dot{\underline{e}} = A_c \underline{e} + \underline{b}_c \{[f(\underline{x}) - \hat{f}(\underline{x};\underline{\theta}_f)] + [g(\underline{x}) - \hat{g}(\underline{x};\underline{\theta}_g)]u_c\} \tag{8.9}$$

where

$$A_c = \begin{bmatrix} 0 & 1 \\ -k_2 & -k_1 \end{bmatrix} \text{ and } \underline{b}_c = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{8.10}$$

A_c is a stable matrix, because $|sI - A_c| = s^2 + k_1 s + k_2$ has its roots in the open left plane. Therefore, there exists a unique symmetric positive definite 2×2 matrix P which satisfies the Lyapunov equation [198]:

$$A_c^T P + P A_c = -Q \tag{8.11}$$

where Q is an arbitrary 2×2 positive definite matrix.

The next task is to replace $\hat{f}$ and $\hat{g}$ by FLSs represented by the following equations:

$$\begin{aligned} \hat{f}(\underline{x};\underline{\theta}_f) &= \sum_{l=1}^{M} \theta_{fl}\xi_l(\underline{x}) \\ &= \underline{\theta}_f{}^T \underline{\xi}(\underline{x}) \end{aligned} \tag{8.12}$$

$$\begin{aligned} \hat{g}(\underline{x};\underline{\theta}_g) &= \sum_{l=1}^{M} \theta_{gl}\xi_l(\underline{x}) \\ &= \underline{\theta}_g{}^T \underline{\xi}(\underline{x}) \end{aligned} \tag{8.13}$$

where $\underline{\theta}_f = [\theta_{f1}, \ldots, \theta_{fM}]^T$, $\underline{\theta}_g = [\theta_{g1}, \ldots, \theta_{gM}]^T$ and $\underline{\xi}(\underline{x}) = [\xi_1(\underline{x}), \ldots, \xi_M(\underline{x})]^T$. $\xi_l(\boldsymbol{x})$ are called *fuzzy basis functions* (FBFs) (refer to Section 2.4) and are given by

$$\xi_l(\underline{x}) = \frac{\prod_{i=1}^{p} \mu_{F_i^l}(x_i)}{\sum_{l=1}^{M} \prod_{i=1}^{p} \mu_{F_i^l}(x_i)} \tag{8.14}$$

where $l = 1, 2, ..., M$ (M is the number of rules in the FLS) and p is the number of inputs.

An adaptive law should be developed to adjust the parameters in the FLSs in order to force the tracking error to converge to zero. Define the optimal parameter vectors for $\hat{f}$ and $\hat{g}$ as

$$\underline{\theta}_f^* \triangleq \arg min_{\underline{\theta}_f \in \Omega_f}[sup_{\underline{x}}|f(x) - \hat{f}(\underline{x};\underline{\theta}_f)|] \tag{8.15}$$

$$\underline{\theta}_g^* \triangleq \arg min_{\underline{\theta}_g \in \Omega_g}[sup_{\underline{x}}|g(x) - \hat{g}(\underline{x};\underline{\theta}_g)|] \tag{8.16}$$

where Ω_f and Ω_g are constraint sets for $\underline{\theta}_f$ and $\underline{\theta}_g$, respectively, specified by the designer. $\underline{\theta}_f$ and $\underline{\theta}_g$ should be bounded. Furthermore, $\hat{g}(\underline{x};\underline{\theta}_g)$ should be a negative function. Therefore Ω_f and Ω_g are defined as

$$\Omega_f = \{\underline{\theta}_f : \|\underline{\theta}_f\| \leq M_f\} \tag{8.17}$$

$$\Omega_g = \{\underline{\theta}_g : \|\underline{\theta}_g\| \leq M_g, \theta_{gl} \leq -\varepsilon\} \tag{8.18}$$

where M_f, M_g and ε are positive constants specified by the designer.

Now define the *minimum approximation error* as

$$w = [f(\underline{x}) - \hat{f}(\underline{x};\underline{\theta}_f^*)] + [g(\underline{x}) - \hat{g}(\underline{x};\underline{\theta}_g^*)]u_c \tag{8.19}$$

Then the error equation (8.9) can be written as

$$\dot{\underline{e}} = A_c \underline{e} + \underline{b}_c \{[\hat{f}(\underline{x};\underline{\theta}_f^*) - \hat{f}(\underline{x};\underline{\theta}_f)] + [\hat{g}(\underline{x};\underline{\theta}_g^*) - \hat{g}(\underline{x};\underline{\theta}_g)]u_c + w\} \tag{8.20}$$

If $\hat{f}$ and $\hat{g}$ are in the form of equations (8.12) and (8.13) then

$$\dot{\underline{e}} = A_c \underline{e} + \underline{b}_c \underline{\phi}_f^T \underline{\xi}(\underline{x}) + \underline{b}_c \underline{\phi}_g^T \underline{\xi}(\underline{x}) u_c + \underline{b}_c w \tag{8.21}$$

where

$$\underline{\phi}_f = \underline{\theta}_f^* - \underline{\theta}_f \tag{8.22}$$

$$\underline{\phi}_g = \underline{\theta}_g^* - \underline{\theta}_g \tag{8.23}$$

Consider the Lyapunov function candidate

$$V \triangleq \frac{1}{2}\underline{e}^T P \underline{e} + \frac{1}{2\gamma_1}\underline{\phi}_f^T \underline{\phi}_f + \frac{1}{2\gamma_2}\underline{\phi}_g^T \underline{\phi}_g \tag{8.24}$$

where γ_1 and γ_2 are positive constants which will be used as the learning rate in the adaptation procedure. Since $\underline{\phi}_f$ is a vector, $\underline{\phi}_f^T \dot{\underline{\phi}}_f$ has only one element, therefore, $\underline{\phi}_f^T \dot{\underline{\phi}}_f = (\underline{\phi}_f^T \dot{\underline{\phi}}_f)^T = \dot{\underline{\phi}}_f^T \underline{\phi}_f$. The same is true about $\underline{\phi}_g$. Using this and equations (8.21) and (8.24) it can be shown that

$$\dot{V} = -\frac{1}{2}\underline{e}^T Q\underline{e} + \underline{e}^T P\underline{b}_c w\frac{\underline{\phi}_f^T}{\gamma_1}\left[\gamma_1 \underline{e}^T P\underline{b}_c \underline{\xi}(\underline{x}) + \dot{\underline{\phi}}_f\right] + \frac{\underline{\phi}_g^T}{\gamma_2}\left[\gamma_2 \underline{e}^T P\underline{b}_c \underline{\xi}(\underline{x})u_c + \dot{\underline{\phi}}_g\right] \quad (8.25)$$

If $\dot{\underline{\phi}}_f$ and $\dot{\underline{\phi}}_g$ are calculated according to $\dot{\underline{\phi}}_f = -\gamma_1 \underline{e}^T P\underline{b}_c \underline{\xi}(\underline{x})$ and $\dot{\underline{\phi}}_g = -\gamma_2 \underline{e}^T P\underline{b}_c \underline{\xi}(\underline{x})u_c$, then

$$\dot{V} = -\frac{1}{2}\underline{e}^T Q\underline{e} + \underline{e}^T P\underline{b}_c w \quad (8.26)$$

w is proportional to the difference between the unknown functions $f(\underline{x})$ and $g(\underline{x})$ and the output of the FLSs when the optimal parameter vectors are used. It is expected that $\underline{e}^T P\underline{b}_c w$, which is of the order of the minimum approximation error, has a small value such that $\frac{1}{2}\underline{e}^T Q\underline{e} > \left|\underline{e}^T P\underline{b}_c w\right|$ and as a result

$$\dot{V} < 0 \quad (8.27)$$

Having achieved equation (8.27), $\underline{e}$, $\underline{\phi}_f$ and $\underline{\phi}_g$ will all converge to zero.

From equations (8.22) and (8.23) $\dot{\underline{\theta}}_f = -\dot{\underline{\phi}}_f$ and $\dot{\underline{\theta}}_g = -\dot{\underline{\phi}}_g$, therefore, the adaptation law for the parameter vectors of the FLSs are

$$\dot{\underline{\theta}}_f = \gamma_1 \underline{e}^T P\underline{b}_c \underline{\xi}(\underline{x}) \quad (8.28)$$

$$\dot{\underline{\theta}}_g = \gamma_2 \underline{e}^T P \underline{b}_c \underline{\xi}(\underline{x}) u_c \tag{8.29}$$

The final problem is how to achieve the constraints represented by equations (8.17) and (8.18). The adaptation laws (8.28) and (8.29) cannot guarantee these constraints. To solve this problem, the parameter projection algorithm [200] can be used. If the parameter vector is within the constraint set or on the boundaries of the constraint set but moving toward the inside of the constraint set, then the simple adaptation laws (8.28) and (8.29) will be used. Otherwise, if the parameter vector is on the boundary of the constraint set and moving toward the outside of the constraint set, the projection algorithm will be used to modify the adaptation law as

$$\dot{\underline{\theta}}_f = \gamma_1 \underline{e}^T P \underline{b}_c \underline{\xi}(\underline{x}) - \gamma_1 \underline{e}^T P \underline{b}_c \frac{\underline{\theta}_f \underline{\theta}_f^T \underline{\xi}(\underline{x})}{\left|\underline{\theta}_f\right|^2} \tag{8.30}$$

$$\dot{\underline{\theta}}_g = \gamma_2 \underline{e}^T P \underline{b}_c \underline{\xi}(\underline{x}) u_c - \gamma_2 \underline{e}^T P \underline{b}_c u_c \frac{\underline{\theta}_g \underline{\theta}_g^T \underline{\xi}(\underline{x})}{\left|\underline{\theta}_g\right|^2} \tag{8.31}$$

8.4.1 The Desired Track

The desired track is the output of the predictor designed in Section 7.6.1 as given by equation (7.10). The output of this predictor will be compared to the output of the plant at each sampling time to get the error.

8.4.2 Design procedure

Step 1: Off-line Preprocessing

❶ Specify $k_1 = 1, k_2 = 4$ such that the roots of $s^2 + k_1 s + k_2 = 0$ are $s_1 = -0.5 - j\sqrt{3}/2$ and $s_2 = -0.5 + j\sqrt{3}/2$. Other values for k_1 and k_2 are possible as long as roots s_1 and s_2 are in a position in the left half-plane with good damping characteristics.

❷ Specify a diagonal positive definite 2×2 matrix Q as $Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. There is a trade off between elements of Q and the learning rates γ_1 and γ_2. For bigger elements of Q, a smaller value should be chosen for learning rates γ_1 and γ_2.

❸ Solve the Lyapunov equation (8.11) to obtain a symmetric positive definite matrix P.

❹ Specify the design parameters γ_1, γ_2, M_f, M_g and ε based on the practical constraints. These values have been chosen to be $\gamma_1 = 2$, $\gamma_2 = 20$, $M_f = 3$ $M_g = 100$, and $\varepsilon = 0.5$. There is a compromise in choosing these parameters. Small learning rates will result in a less adaptable system which is more stable, compared with a system with higher learning rates. M_f and M_g are the upper limits on $\|\underline{\theta}_f\|$ and $\|\underline{\theta}_g\|$. The lower limit on the absolute value of the elements of the parameter vector $\underline{\theta}_g$ is determined by ε.

Step 2: Initial Fuzzy Logic PSS Construction

❶ Define m_1 fuzzy sets $F_1^{l_1}$ for input $\Delta\omega$ where $l_1 = 1, 2, \ldots, m_1$ and m_2 fuzzy sets $F_2^{l_2}$ for input ΔP where $l_2 = 1, 2, \ldots, m_2$. Here m_1 and m_2 are chosen to be $m_1 = m_2 = 5$. As stated in Chapter 5, more fuzzy sets will result in a more smooth response. However, this will make the controller more complicated in terms of computational requirements. five fuzzy sets for each input is sufficient for the PSS to be designed. The fuzzy sets for input $\Delta\omega$ are defined according to the membership functions shown in Fig. 8.2 and the fuzzy sets for input ΔP are defined according to the membership functions shown in Fig. 8.3. The ranges of membership functions are chosen according to what is expected for maximum and minimum of speed deviation and accelerating power. For the shape of membership functions, Gaussian functions have been chosen to make the calculations easier. The membership functions labelled ZR for inputs x_1 and x_2 have been chosen to

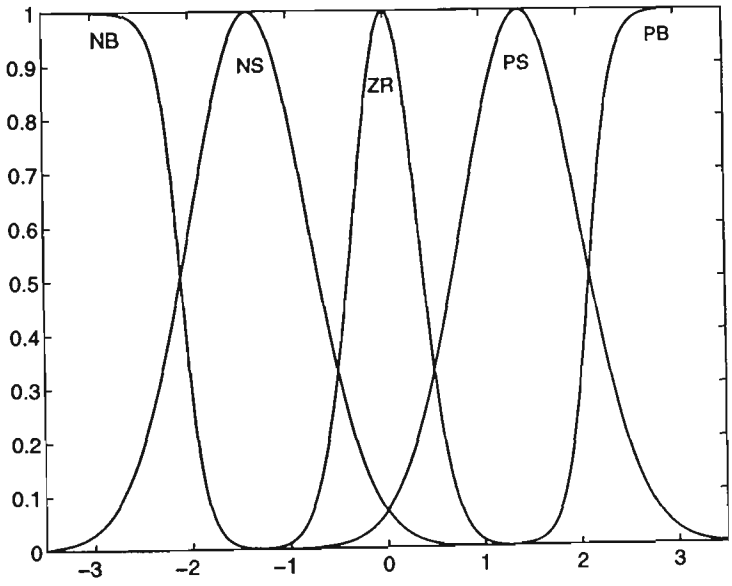


Fig. 8.2 Fuzzy membership functions for input x_1

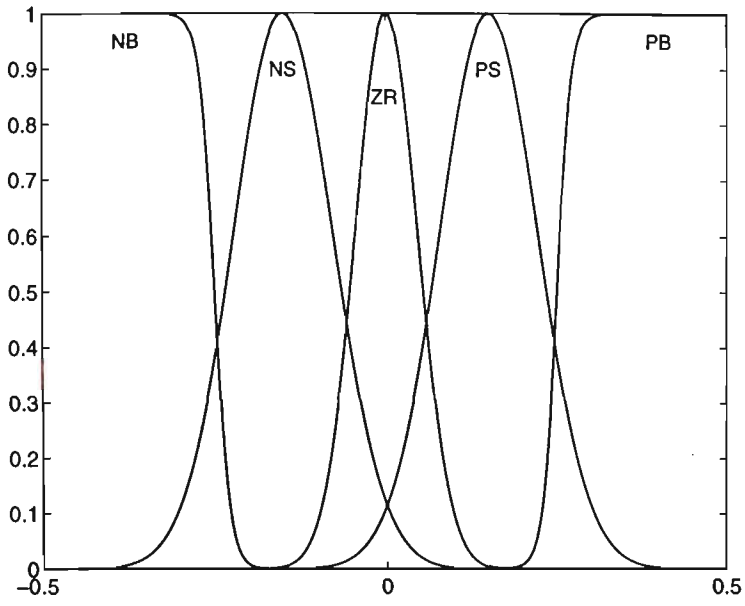


Fig. 8.3 Fuzzy membership functions for input x_2

be narrower than other labels. This will cause the oscillations in speed deviation response to damp faster when the speed deviation has reduced to small values around zero with a small acceleration.

❷ Construct the fuzzy basis functions from the input membership functions as

$$\xi^{(l_1, l_2)}(\underline{x}) = \frac{\mu_{F_1^{l_1}}(x_1)\mu_{F_2^{l_2}}(x_2)}{\sum_{l_1=1}^{m_1}\sum_{l_2=1}^{m_2}\mu_{F_1^{l_1}}(x_1)\mu_{F_2^{l_2}}(x_2)} \tag{8.32}$$

and collect them into a $m_1 \times m_2$-dimensional vector $\underline{\xi}(\underline{x})$ in a natural ordering for $l_1 = 1, 2, \ldots, m_1$ and $l_2 = 1, 2, \ldots, m_2$.

❸ Construct the fuzzy rule base for the FLSs $\hat{f}(\underline{x};\underline{\theta}_f)$ and $\hat{g}(\underline{x};\underline{\theta}_g)$ which consist of $m_1 \times m_2$ rules whose IF parts comprise of all the possible

combinations of the fuzzy input sets. Specifically, the fuzzy rule base for $\hat{g}(\underline{x};\underline{\theta}_g)$ consists of rules:

$$R_g^{(l_1, l_2)}: \text{IF } x_1 \text{ is } F_1^{l_1} \text{ and } x_2 \text{ is } F_2^{l_2}, \text{ THEN } \hat{g}(\underline{x};\underline{\theta}_g) \text{ is } \chi^{(l_1, l_2)} \tag{8.33}$$

where $\chi^{(l_1, l_2)}$ is the centre of gravity of the fuzzy output set. Construct vector $\underline{\theta}_g$ as a collection of the values of $\chi^{(l_1, l_2)}$ in the same ordering as $\underline{\xi}(\underline{x})$. For example, for the specific synchronous machine under study, the fuzzy rule base and the parameter vector $\underline{\theta}_g$ can be constructed from Table 8.3. As an example, the eighth rule is

Table 8.3. Decision table constructed of 25 rules

Speed Deviation ($\Delta\omega$)	Accelerating Power (ΔP)				
	NB	NS	ZR	PS	PB
NB	-6	-12	-18	-12	-6
NS	-12	-18	-24	-18	-12
ZR	-18	-24	-30	-24	-18
PS	-12	-18	-24	-18	-12
PB	-6	-12	-18	-12	-6

$R_g^{(2, 3)}$: IF $\Delta\omega$ is **Negative Small** and ΔP is **Zero**, THEN $\hat{g}(\underline{x};\underline{\theta}_g)$ should be **close to -24**.

From this rule it is seen that the eighth element of the parameter vector is

$\underline{\theta}_g(8) = \underline{\theta}_g^{(2,3)} = -24$. In this way the 25-element initial parameter vector $\underline{\theta}_g$ will be obtained.

Since there is not enough information about $\hat{f}(\underline{x};\underline{\theta}_f)$, the initial value of $\underline{\theta}_f$ is chosen to be zero.

❹ From steps ❷ and ❸ $\hat{f}(\underline{x};\underline{\theta}_f)$ and $\hat{g}(\underline{x};\underline{\theta}_g)$ are obtained as

$$\hat{f}(\underline{x};\underline{\theta}_f) = \underline{\theta}_f^T \underline{\xi}(\underline{x}) \tag{8.34}$$

$$\hat{g}(\underline{x};\underline{\theta}_g) = \underline{\theta}_g^T \underline{\xi}(\underline{x}) \tag{8.35}$$

Then the controlling signal will be obtained from equation (8.7).

Step 3: On-Line Adaptation

❶ Apply the controlling signal to the plant.

❷ Use the adaptive laws explained in the previous section to adjust the parameter vectors $\underline{\theta}_f$ and $\underline{\theta}_g$.

8.4.3 Simulation results

The performance of the indirect AFPSS is evaluated by firstly simulating the response of the system with the initial FPSS when $\underline{\theta}_g$ is chosen randomly. Then, $\underline{\theta}_g$ is chosen according to Table 8.3, but there is no on-line adaptation. Finally, the improved performance with on-line adaptation is examined. The simulations have been done for the normal operating condition, i.e., $P_o = 0.9$ pu with a power factor

of 0.9 lagging and E_B = 1.0 pu. The transmission line parameters are R_e = 0.03 pu and X_e = 0.15 pu.

The speed deviation response of the system for the first case, i.e., random selection of $\underline{\theta}_g$ without on-line adaptation is shown in Fig. 8.4. The stabiliser output is shown in Fig. 8.5. As seen from the figures, the stabilising signal and the speed deviation response are unstable. In this case the performance index is J_p = 1348 which is calculated over the time period of 4 seconds. The response of the indirect AFPSS when $\underline{\theta}_g$ is chosen according to Table 8.3, but there is no on-line adaptation, is shown in Fig. 8.6. In this case J_p = 64.5. The output of the indirect AFPSS in this case is shown in Fig. 8.7.

The response of the indirect AFPSS with on-line adaptation is shown in Fig. 8.8. In this case J_p = 47.1, which shows a better performance compared to the

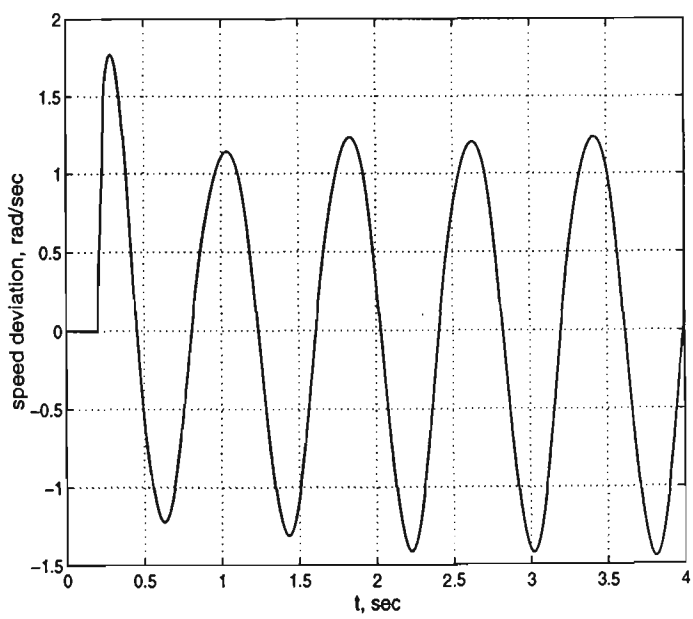


Fig. 8.4 Response of the system with initial indirect FPSS without on-line adaptation

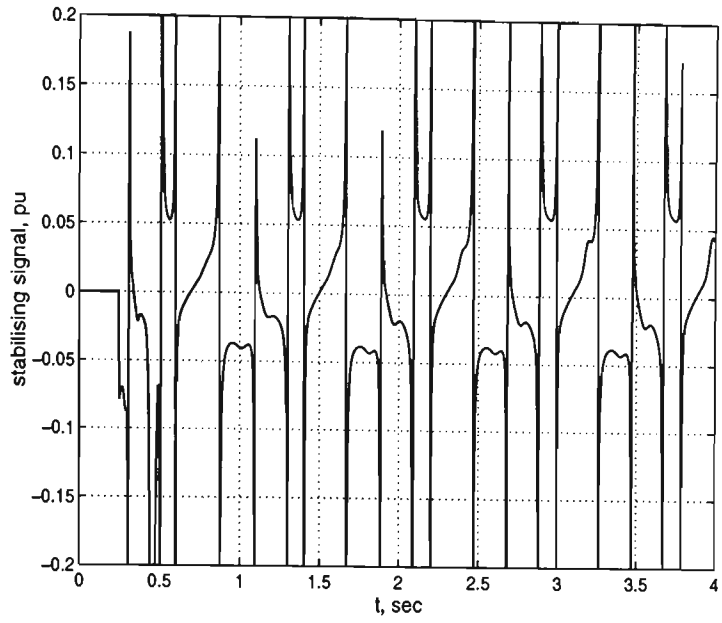


Fig. 8.5 Output of the initial indirect FPSS output without on-line adaptation

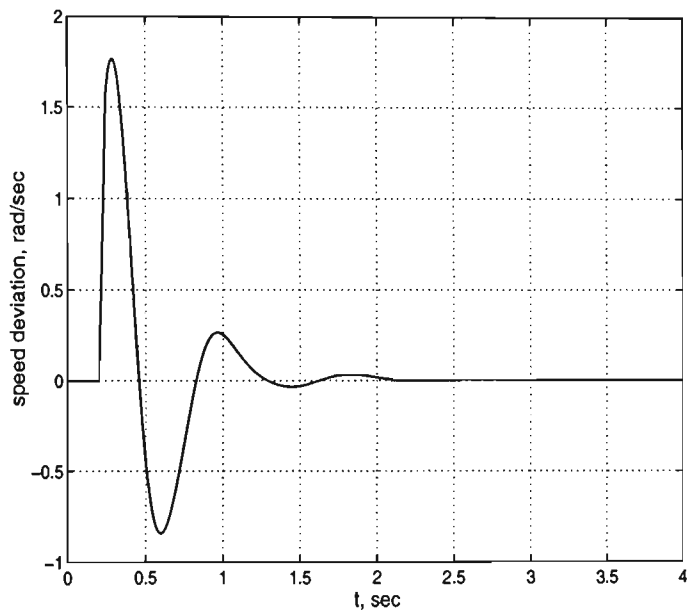


Fig. 8.6 Response of the system with indirect AFPSS without on-line adaptation for normal operating conditions

previous case. The stabilising signal for this case is shown in Fig. 8.9.

In order to shed more light on the better performance of the indirect AFPSS

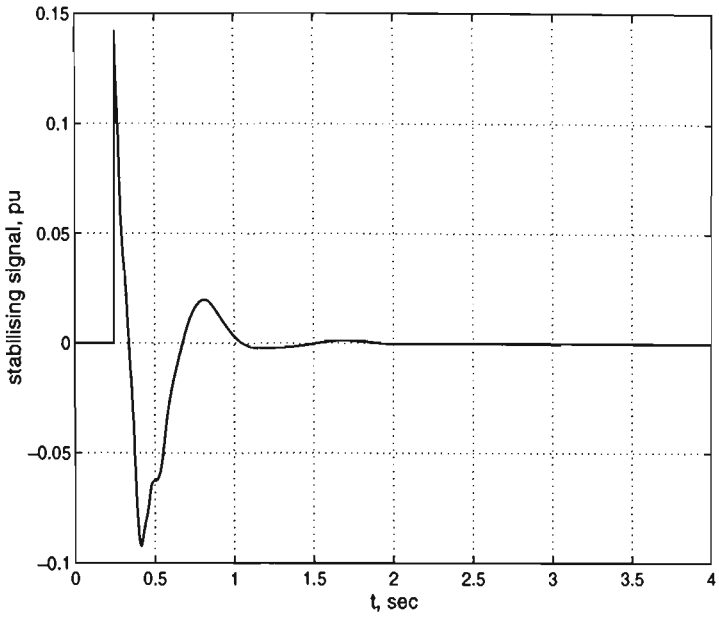


Fig. 8.7 The indirect FPSS output without on-line adaptation for the normal operating conditions

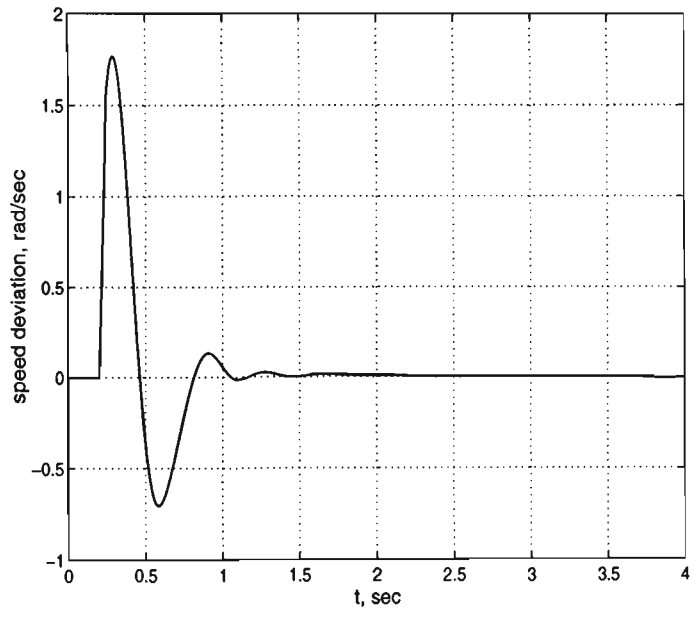


Fig. 8.8 Response of the system with indirect AFPSS with on-line adaptation

compared to the initial FPSS without adaptation, lets change the operating condition and the line impedance. The new operating condition is $P_o = 0.7$ pu with a power

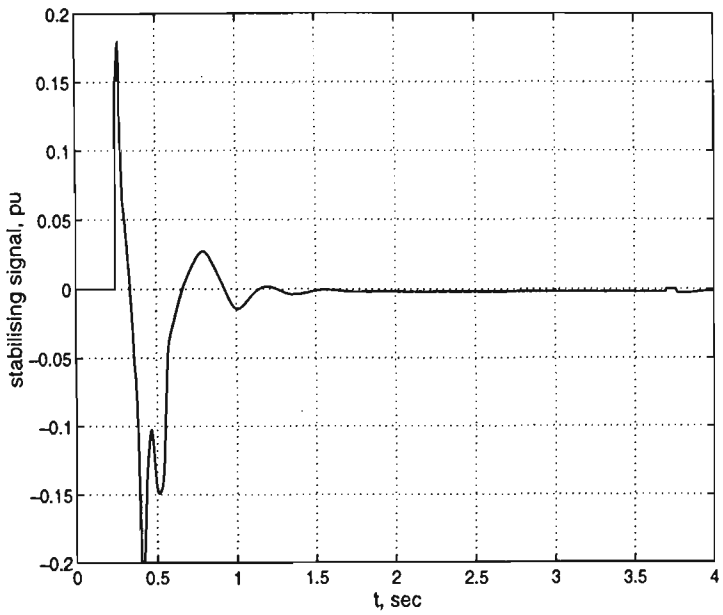


Fig. 8.9 The indirect AFPSS output with on-line adaptation

factor of 0.7 lagging and E_B = 1.0 pu. The transmission line parameters are R_e = 0.05 pu and X_e = 0.45 pu. The response of the initial FPSS, with $\underline{\theta}_g$ chosen according to Table 8.3, without on-line adaptation is shown in Fig. 8.10. In this case J_p = 397. The stabilising signal is shown in Fig. 8.11. Now, the response of the indirect AFPSS with on-line adaptation is obtained as shown in Fig. 8.12. In this case J_p = 149, which shows a lot of improvement compared to the initial FPSS result. However, the response is not optimum (J_p has not the minimum obtainable value). The output of the indirect AFPSS in this case is shown in Fig. 8.13.

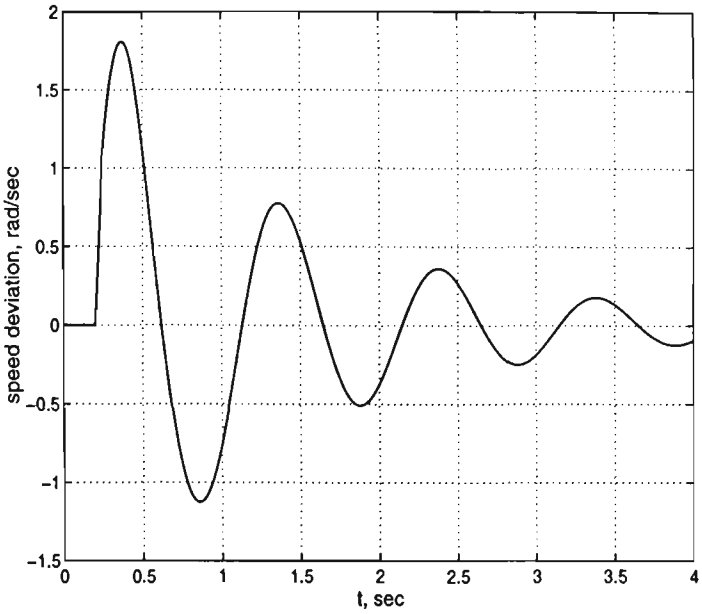


Fig. 8.10 Response of the system with the indirect AFPSS without on-line adaptation for the new operating condition and line impedance

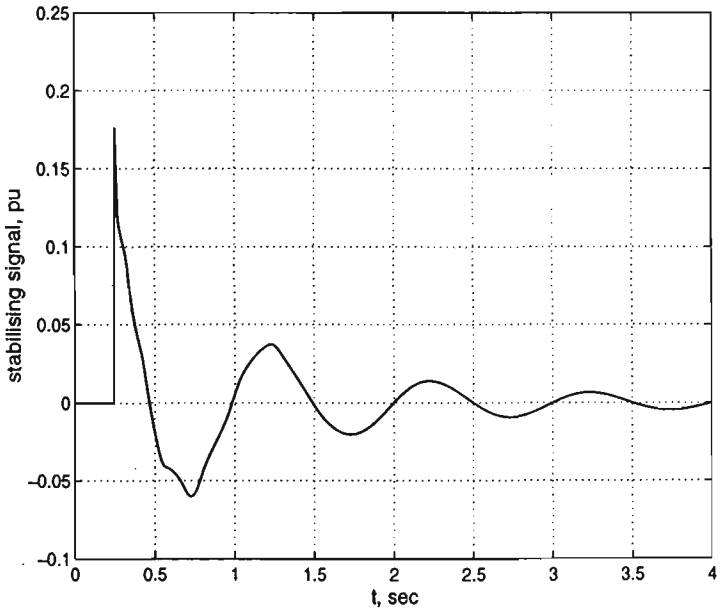


Fig. 8.11 The output of the initial indirect AFPSS without on-line adaptation for the new operating condition and line impedance

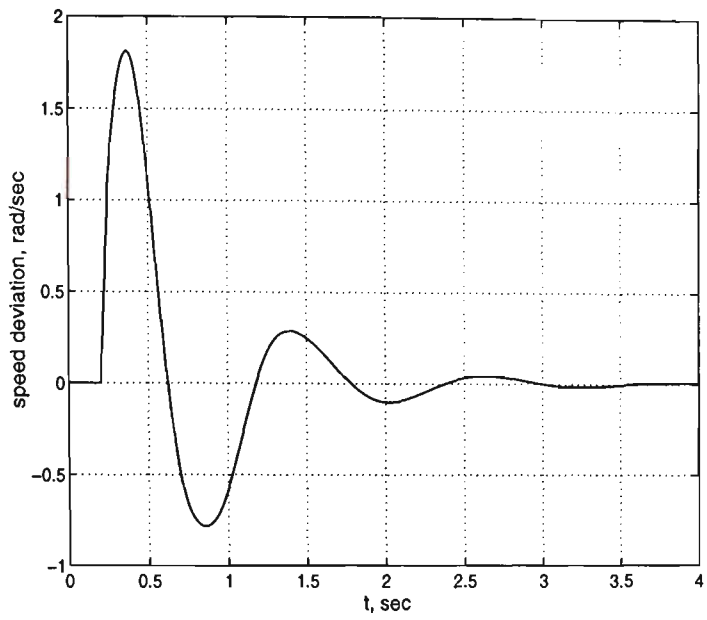


Fig. 8.12 Response of the system with the indirect AFPSS with on-line adaptation for the new operating condition and line impedance

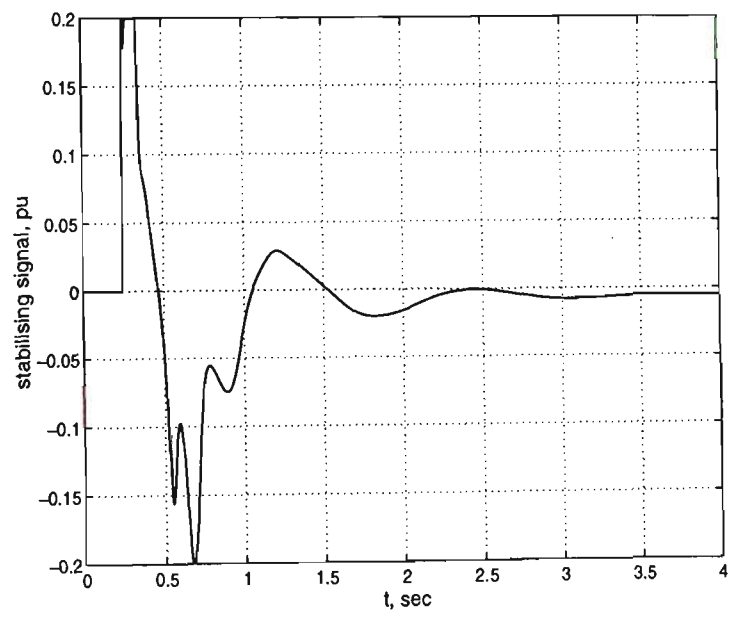


Fig. 8.13 The output of the indirect AFPSS with on-line adaptation for the new operating condition and line impedance

8.5 Direct Adaptive Fuzzy Logic PSS

In this section the procedure of designing a direct AFPSS for a cogenerator system is explained. The system is represented by equation (8.1). The control objective is to force y to follow a given desired signal y_m, under the constraint that all signals involved must be bounded. More specifically, the control objective is [195] to determine a feedback control $u = u(\underline{x};\underline{\theta})$ (based on fuzzy logic systems) and an adaptive law for adjusting the parameter vector $\underline{\theta}$ such that:

1. The closed-loop system be globally stable in the sense that all variables, $\underline{x}(t)$, $\underline{\theta}$, and $u(\underline{x};\underline{\theta})$ must be bounded; that is, $\|\underline{x}\| \le M_x$, $\|\underline{\theta}\| \le M_\theta$, and $|u| \le M_u$ for all $t \ge 0$, where M_x, M_θ, and M_u are design parameters specified by the designer.

2. The tracking error, $e = y - y_m$, should be as small as possible under the constraints in the previous objective.

In this section, the objective is to design a fuzzy logic controller with a control law u close enough to u^* (refer to equation (8.5)).

Similar to the procedure followed in the previous section, the adaptation law is derived by the following procedure. As in the previous case this approach in designing adaptive direct fuzzy logic controllers was first introduced by Wang [195]. However, the application of this approach in designing direct adaptive fuzzy logic PSSs is the contribution of this thesis.

By adding $g(\underline{x})u^*$ to both sides of equation (8.5) and using equation (8.6), the following equation is derived:

$$\begin{aligned} \ddot{y} + g(\underline{x})u^* &= f(\underline{x}) + g(\underline{x})u + [-f(\underline{x}) + \ddot{y}_m - \underline{k}^T \underline{e}] \\ &= g(\underline{x})u + \ddot{y}_m - \underline{k}^T \underline{e} \end{aligned} \tag{8.36}$$

or

$$\ddot{e} = -\underline{k}^T \underline{e} + g(\underline{x})(u - u^*) \tag{8.37}$$

Since $\dot{\underline{e}} = [\dot{e}, \ddot{e}]^T$ equation (8.37) can be written as

$$\dot{\underline{e}} = A_c \underline{e} + \underline{g}_c(\underline{x})(u - u^*) \tag{8.38}$$

where

$$A_c = \begin{bmatrix} 0 & 1 \\ -k_2 & -k_1 \end{bmatrix} \text{ and } \underline{g}_c(\underline{x}) = \begin{bmatrix} 0 \\ g(\underline{x}) \end{bmatrix} \tag{8.39}$$

A_c is a stable matrix, because $|sI - A_c| = s^2 + k_1 s + k_2$ has its roots in the open left plane. Therefore, there exists a unique symmetric positive definite (2×2) matrix P which satisfies the Lyapunov equation:

$$A_c^T P + P A_c = -Q \tag{8.40}$$

where Q is an arbitrary 2×2 diagonal positive definite matrix.

Now, consider u_c as the output of a FLS which can be represented as

$$\begin{aligned} u_c(\underline{x};\underline{\theta}) &= \sum_{l=1}^{M} \theta_l \xi_l(\underline{x}) \\ &= \underline{\theta}^T \underline{\xi}(\underline{x}) \end{aligned} \tag{8.41}$$

where $\underline{\theta} = [\theta_1, \ldots, \theta_M]^T$ and $\underline{\xi}(\underline{x}) = [\xi_1(\underline{x}), \ldots, \xi_M(\underline{x})]^T$ .In order to develop an adaptive law to adjust the parameter vector $\underline{\theta}$ such that $u_c(\underline{x};\underline{\theta})$ is close enough to u^* , define the optimal parameter vector

$$\underline{\theta}^* \triangleq \arg min_{\underline{\theta}}[sup_{\underline{x}}|u^* - u_c(\underline{x};\underline{\theta})|] \tag{8.42}$$

and the minimum approximation error

$$w \triangleq u^* - u_c(\underline{x};\underline{\theta}^*) \tag{8.43}$$

The error equation (8.38) can be rewritten as

$$\begin{aligned} \dot{\underline{e}} &= A_c \underline{e} + \underline{g}_c(\underline{x})[u_c(\underline{x};\underline{\theta}) - u_c(\underline{x};\underline{\theta}^*) - w] \\ &= A_c \underline{e} + \underline{g}_c(\underline{x})[u_c(\underline{x};\underline{\theta}) - u_c(\underline{x};\underline{\theta}^*)] - \underline{g}_c(\underline{x}) w \end{aligned} \tag{8.44}$$

Since $u_c(\underline{x};\underline{\theta}) = \underline{\theta}^T \underline{\xi}(\underline{x})$ and $u_c(\underline{x};\underline{\theta}^*) = \underline{\theta}^{*T} \underline{\xi}(\underline{x})$, (8.44) can be written as

$$\dot{\underline{e}} = A_c\underline{e} + \underline{g}_c(\underline{x})\underline{\phi}^T\underline{\xi}(\underline{x}) - \underline{g}_c(\underline{x})w \tag{8.45}$$

where

$$\underline{\phi} = \underline{\theta} - \underline{\theta}^* \tag{8.46}$$

Define the Lyapunov function candidate

$$V \triangleq \frac{1}{2}\underline{e}^T P\underline{e} + \frac{1}{2\gamma}\underline{\phi}^T\underline{\phi} \tag{8.47}$$

where γ is a constant which will be used as the learning rate in the adaptation procedure. Since $\underline{\phi}^T\dot{\underline{\phi}}$ has only one element (a 1×1 matrix), $\underline{\phi}^T\dot{\underline{\phi}} = (\underline{\phi}^T\dot{\underline{\phi}})^T = \dot{\underline{\phi}}^T\underline{\phi}$ and using (8.45) it can be shown that

$$\begin{aligned} \dot{V} &= -\frac{1}{2}\underline{e}^T Q\underline{e} + \underline{e}^T P\underline{g}_c(\underline{x})\underline{\phi}^T\underline{\xi}(\underline{x}) - \underline{e}^T P\underline{g}_c(\underline{x})w + \frac{1}{\gamma}\underline{\phi}^T\dot{\underline{\phi}} \\ &= -\frac{1}{2}\underline{e}^T Q\underline{e} + \underline{e}^T\underline{P}_n g(\underline{x})\underline{\phi}^T\underline{\xi}(\underline{x}) + \frac{1}{\gamma}\underline{\phi}^T\dot{\underline{\phi}} - \underline{e}^T\underline{P}_n g(\underline{x})w \end{aligned} \tag{8.48}$$

where $\underline{P}_n$ is the last column of P.

Equivalently

$$\dot{V} = -\frac{1}{2}\underline{e}^T Q\underline{e} + \frac{\underline{\phi}^T}{\gamma}[\gamma\underline{e}^T\underline{P}_n g(\underline{x})\underline{\xi}(\underline{x}) + \dot{\underline{\phi}}] - \underline{e}^T\underline{P}_n g(\underline{x})w \tag{8.49}$$

If $\dot{\underline{\phi}}$ is chosen such that $\dot{\underline{\phi}} = -\gamma \underline{e}^T \underline{P}_n g(\underline{x}) \underline{\xi}(\underline{x})$, then

$$\dot{V} = -\frac{1}{2}\underline{e}^T Q \underline{e} - \underline{e}^T \underline{P}_n g(\underline{x}) w \tag{8.50}$$

w is the difference between the ideal controlling signal u^* and the output of the FLC when the optimal parameter vector is used, i.e., $u_c(\underline{x};\underline{\theta}^*)$. It is expected that $\underline{e}^T \underline{P}_n g(\underline{x}) w$ which is of the order of the minimum approximation error has a small value such that $\frac{1}{2}\underline{e}^T Q \underline{e} > \left|\underline{e}^T \underline{P}_n g(\underline{x}) w\right|$ and as a result

$$\dot{V} < 0 \tag{8.51}$$

From equation (8.46) $\dot{\underline{\phi}} = \dot{\underline{\theta}}$, therefore the adaptation law is

$$\dot{\underline{\theta}} = -\gamma \underline{e}^T \underline{P}_n g(\underline{x}) \underline{\xi}(\underline{x}) \tag{8.52}$$

If the normalised learning rate is defined to be $\gamma_n \triangleq -\gamma g(\underline{x})$ which is a positive number then

$$\dot{\underline{\theta}} = \gamma_n \underline{e}^T \underline{P}_n \underline{\xi}(\underline{x}) \tag{8.53}$$

γ_n will be specified by the designer. Since $g(\underline{x})$ is unknown, the value of $-\gamma g(\underline{x})$ can

not be calculated. However, this will not create any problem. As long as the sign of γ_n is chosen properly (positive in this case), the parameter vector $\underline{\theta}$ will change in the desired direction. The specific value of γ_n depends on the required rate of change of $\underline{\theta}$. γ_n can be chosen to be a constant. A small value of γ_n means less adaptation. A big value of γ_n may cause instability.

The constraint $\|\underline{\theta}\| \leq M_\theta$ is achieved using the parameter projection algorithm. If the parameter vector is within the constraint set or on the boundaries of the constraint set but moving toward the inside of the constraint set, then the simple adaptive law represented by equation (8.53) will be used. Otherwise, if the parameter vector is on the boundary of the constraint set and moving toward the outside of the constraint set, the projection algorithm will be used to modify the adaptive law as follows:

$$\dot{\underline{\theta}} = \gamma_n \underline{e}^T \underline{P}_n \underline{\xi}(\underline{x}) - \gamma_n \underline{e}^T \underline{P}_n \frac{\underline{\theta}\underline{\theta}^T \underline{\xi}(\underline{x})}{|\underline{\theta}|^2} \tag{8.54}$$

The desired track is the same as what was introduced in Section 8.4.1.

8.5.1 Design Procedure

Step 1: Off-line Preprocessing

❶ Specify $k_1 = 1, k_2 = 4$ such that the roots of $s^2 + k_1 s + k_2 = 0$ are $s_1 = -0.5 - j\sqrt{3}/2$ and $s_2 = -0.5 + j\sqrt{3}/2$.

❷ Specify a diagonal positive definite 2×2 matrix Q as $Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. There is a trade off between elements of Q and the learning rate γ_n. For bigger elements of Q, a smaller value should be chosen for learning rate γ_n.

❸ Solve the Lyapunov equation (8.11) to obtain a symmetric positive definite matrix P.

❹ Specify the design parameters $\gamma_n = 1.5$, $M_\theta = 3$, $M_u = 0.2$ and $M_x = 10$ based on the practical constraints.

Step 2: Initial Fuzzy Logic PSS Construction

❶ Define m_1 fuzzy sets $F_1^{l_1}$ for input $x_1 = \Delta\omega$ where $l_1 = 1, 2, \ldots, m_1$ and m_2 fuzzy sets $F_2^{l_2}$ for input $x_2 = (\Delta P)/(2H)$ where $l_2 = 1, 2, \ldots, m_2$. Here m_1 and m_2 are chosen to be $m_1 = m_2 = 5$. The fuzzy sets for input x_1 are defined according to the membership functions shown in Fig. 8.2 and the fuzzy sets for input x_2 are defined according to the membership functions shown in Fig. 8.3.

❷ Construct the fuzzy basis functions from the input membership functions

$$\xi^{(l_1, l_2)}(\underline{x}) = \frac{\mu_{F_1^{l_1}}(x_1)\mu_{F_2^{l_2}}(x_2)}{\sum_{l_1=1}^{m_1}\sum_{l_2=1}^{m_2}\mu_{F_1^{l_1}}(x_1)\mu_{F_2^{l_2}}(x_2)} \tag{8.55}$$

and collect them into a $m_1 \times m_2$-dimensional vector $\underline{\xi}(\underline{x})$ in a natural

ordering for $l_1 = 1, 2, \ldots, m_1$ and $l_2 = 1, 2, \ldots, m_2$.

❸ Construct the fuzzy rule base for the FLS $u_c(\underline{x};\underline{\theta})$ which consists of $m_1 \times m_2$ rules whose IF parts comprise of all the possible combinations of the fuzzy input sets. Specifically, the fuzzy rule base of $u_c(\underline{x};\underline{\theta})$ consists of rules:

$$R^{(l_1, l_2)}: \text{IF } x_1 \text{ is } F_1^{l_1} \text{ and } x_2 \text{ is } F_2^{l_2}, \text{ THEN } u_c \text{ is } G^{(l_1, l_2)} \tag{8.56}$$

where $G^{(l_1, l_2)}$ are the centre of gravity of the fuzzy output sets. Construct vector $\underline{\theta}$ as a collection of the values of $G^{(l_1, l_2)}$ in the same ordering as $\underline{\xi}(\underline{x})$. For example, for the specific synchronous machine under study, the fuzzy rule base and the parameter vector $\underline{\theta}$ can be constructed as follows: the output membership functions are defined in Fig. 8.14. Based on these membership functions and what is known from input-output behaviour of a synchronous machine, decision rules are obtained which are summarized in Table 8.4. The labels for output are then converted to the numbers which represent the centre of gravity of output membership functions as shown in f Table 8.5.

As an example, the ninth rule is

$R^{(2, 4)}$: IF x_1 is **NS** and x_2 is **PS**, THEN u_c should be **close to zero**

From this rule it is seen that the ninth element of the parameter vector is $\underline{\theta}(9) = \theta^{(2, 4)} = 0$. In this way the 25-element initial parameter vector will be obtained.

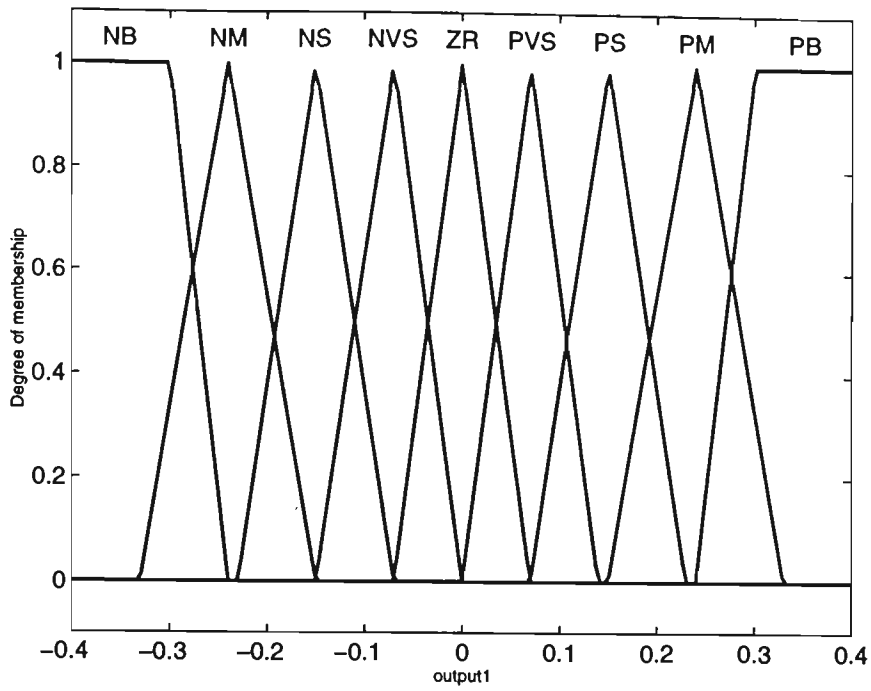


Fig. 8.14 Output membership functions

Table 8.4. Decision table of 25 rules using input and output labels

Speed Deviation ($\Delta\omega$)	Accelerating Power (ΔP)				
	NB	NS	ZR	PS	PB
NB	NB	NM	NS	NVS	ZR
NS	NM	NS	NVS	ZR	PVS
ZR	NS	NVS	ZR	PVS	PS
PS	NVS	ZR	PVS	PS	PM
PB	ZR	PVS	PS	PM	PB

❹ From steps ❷ and ❸ the controlling signal will be obtained as

$$u_c(\underline{x};\underline{\theta}) = \underline{\theta}^T \underline{\xi}(\underline{x}) \tag{8.57}$$

Table 8.5. Decision table of 25 rules using the centre of gravity of the output fuzzy sets

Speed Deviation ($\Delta\omega$)	Accelerating Power (ΔP)				
	NB	NS	ZR	PS	PB
NB	-0.35	-0.24	-0.15	-0.07	0
NS	-0.24	-0.15	-0.07	0	0.07
ZR	-0.15	-0.07	0	0.07	0.15
PS	-0.07	0	0.07	0.15	0.24
PB	0	0.07	0.15	0.24	0.35

Step 3: On-Line Adaptation

❶ Apply the feedback control u_c given by equation (8.57) to the plant.

❷ Use the following adaptive law to adjust the parameter vector $\underline{\theta}$:

$$\dot{\underline{\theta}} = \gamma_n \underline{e}^T \underline{P}_n \underline{\xi}(\underline{x}) - a\gamma_n \underline{e}^T \underline{P}_n \frac{\underline{\theta}\underline{\theta}^T \underline{\xi}(\underline{x})}{|\underline{\theta}|^2} \tag{8.58}$$

where

$$a = \begin{cases} 0, & \text{if } \|\underline{\theta}\| < M_\theta \text{ or } (\|\underline{\theta}\| = M_\theta \text{ and } \underline{e}^T \underline{P}_n \underline{\theta}^T \underline{\xi}(\underline{x}) < 0) \\ 1, & \text{if } (\|\underline{\theta}\| = M_\theta \text{ and } \underline{e}^T \underline{P}_n \underline{\theta}^T \underline{\xi}(\underline{x}) \geq 0) \end{cases} \tag{8.59}$$

8.5.2 Simulation results

Similar simulations to those performed for the indirect AFPSS are performed for the direct AFPSS. Firstly, simulations are done for the normal operating condition, i.e., P_o = 0.9 pu with a power factor of 0.9 lagging and E_B = 1.0 pu with the transmission line parameters R_e = 0.03 pu and X_e = 0.15 pu. The speed deviation response when elements of $\underline{\theta}$ are chosen randomly and there is no on-line adaptation is shown in Fig. 8.15. In this case J_p = 1083. Fig. 8.16 shows the stabilising signal in this case.

Now, with the same conditions, the elements of the parameter vector are chosen according to Table 8.5. The speed deviation response of the system is shown in Fig. 8.17. In this case J_p = 48. The stabilising signal is shown in Fig. 8.18. Since the response is nearly optimum (in the sense that J_p is close to the minimum value obtainable), the system response with on-line adaptation will not have much

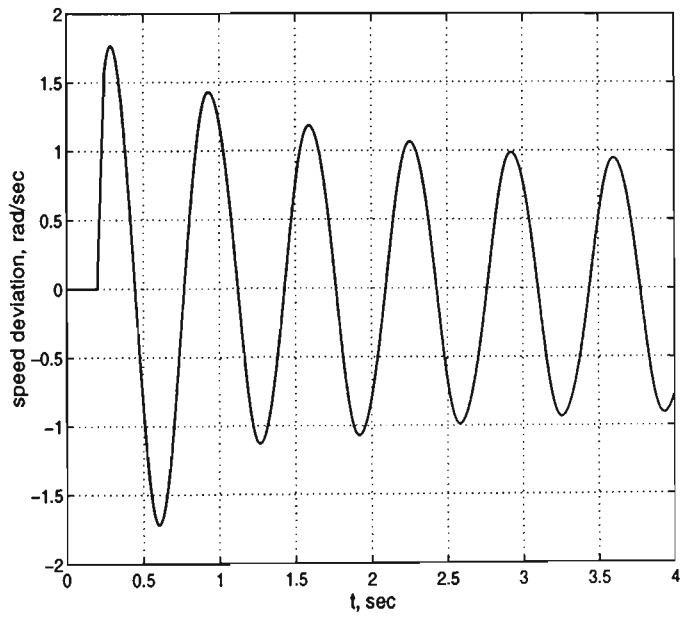


Fig. 8.15 Response of the system with initial direct FPSS without on-line adaptation

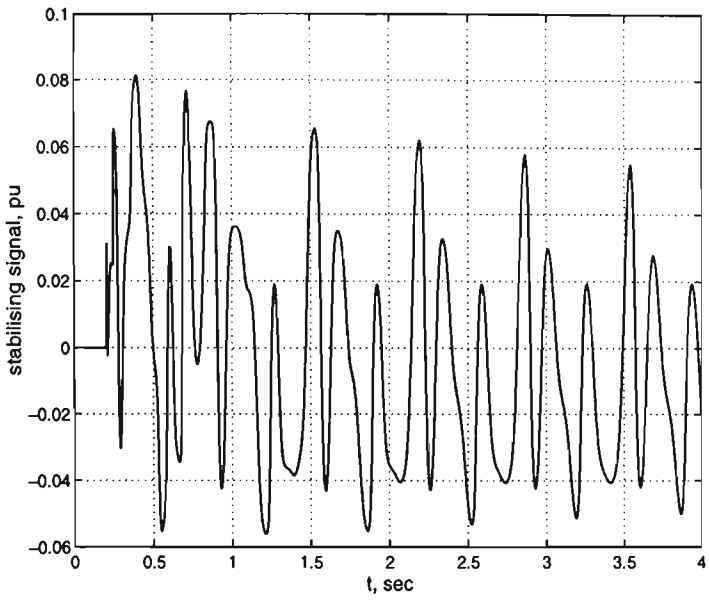


Fig. 8.16 The initial direct FPSS output without on-line adaptation

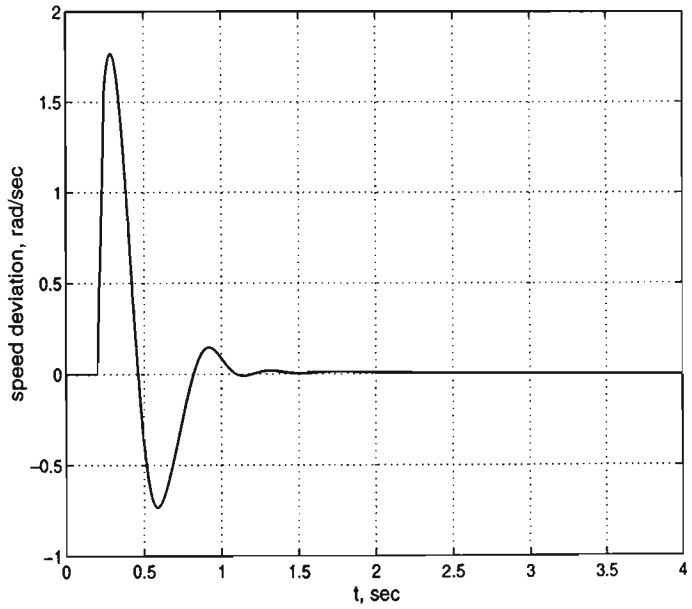


Fig. 8.17 Response of the system with direct FPSS without on-line adaptation for normal operating conditions

difference with the one shown in Fig. 8.17. Therefore, it is not shown here.

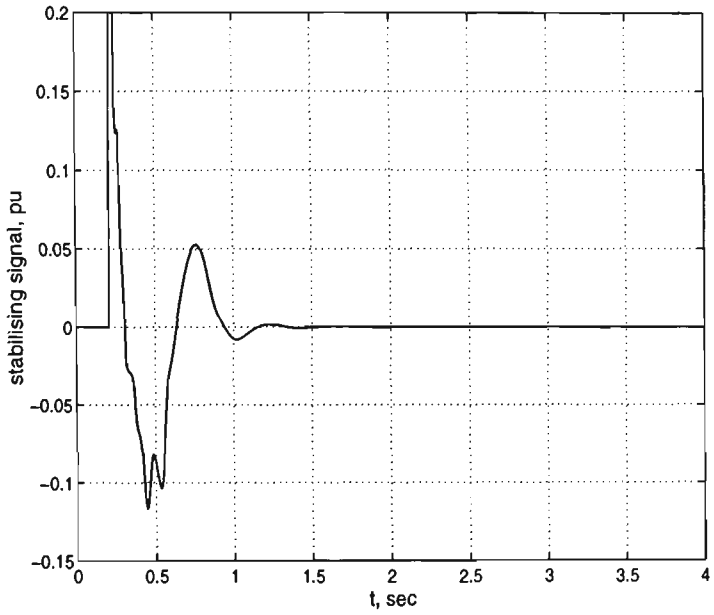


Fig. 8.18 The direct FPSS output without on-line adaptation for the normal operating conditions

For a different operating conditions and line impedances, i.e., P_o = 0.7 pu with a power factor of 0.7 lagging and E_B = 1.0 pu with the transmission line parameters R_e = 0.05 pu and X_e = 0.45 pu, the simulations are repeated. When the parameter vector is chosen according to Table 8.5 but there is no on-line adaptation, the speed deviation response is as shown in Fig. 8.19. In this case J_p = 308. Fig. 8.20 shows the stabilising signal in this case.

The response of the system with direct AFPSS with on-line adaptation for the new conditions is shown in Fig. 8.21. In this case J_p = 123. The stabilising signal is shown in Fig. 8.22. Comparing figures 8.21 and 8.19, it is observed that the adaptability of the direct AFPSS improves the response significantly. Also, comparing Fig. 8.21 to Fig. 8.12 reveals that the direct AFPSS is more effective than the indirect AFPSS. This result seems to be different from what is known for conventional adaptive controllers. In the adaptive control literature, it is well known

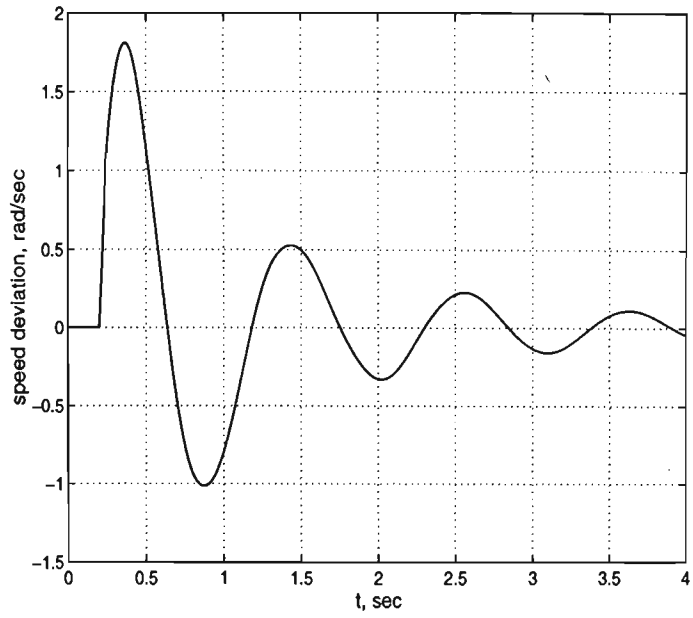


Fig. 8.19 Response of the system with the direct FPSS without on-line adaptation for the new operating condition and line impedance

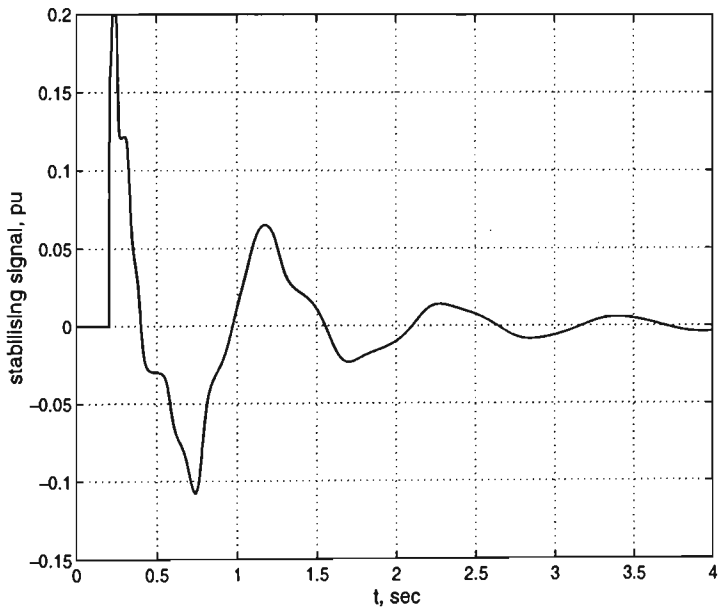


Fig. 8.20 The output of the direct FPSS without on-line adaptation for the new operating condition and line impedance

that the indirect schemes are more robust than direct schemes. The reason for this difference in the results obtained for AFPSSs compared to conventional adaptive

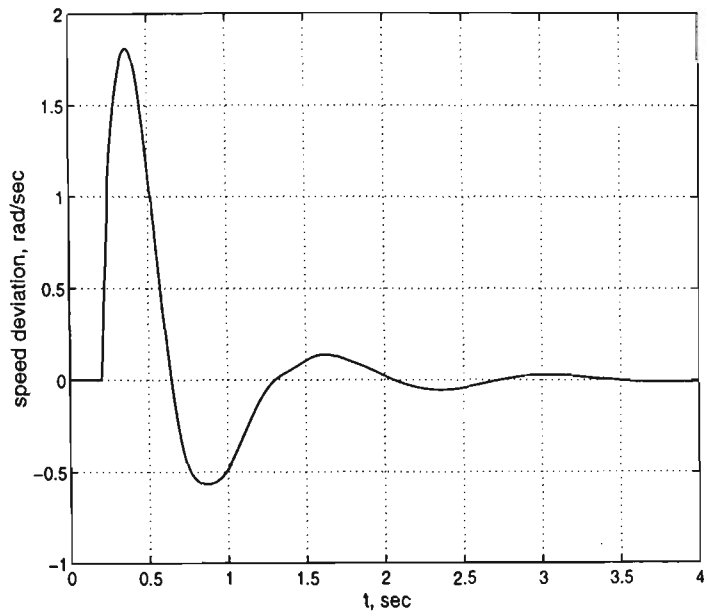


Fig. 8.21 Response of the system with the direct AFPSS with on-line adaptation for the new operating condition and line impedance

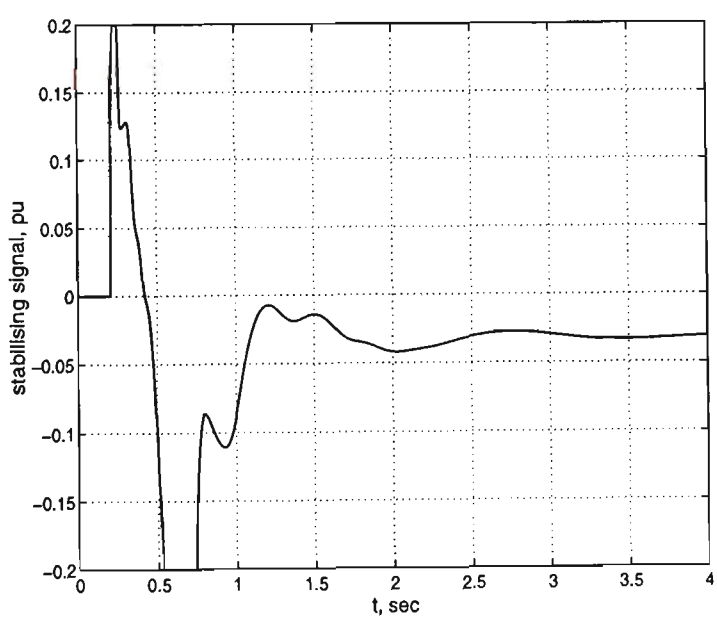


Fig. 8.22 The output of the direct AFPSS with on-line adaptation for the new operating condition and line impedance

schemes can be stated as follows:

in the conventional indirect adaptive schemes the functions such as $f(x)$ and $g(x)$ are assumed to be known. The controller is constructed on the basis of these known functions. On the other hand, for the indirect AFPSS functions $f(x)$ and $g(x)$ are not known. The indirect adaptation scheme in this case is an attempt to estimate these functions and adjust the controller indirectly. This scheme is not as effective as when the functions are known.

8.6 Conclusions

In this chapter two schemes for designing adaptive fuzzy logic power system stabilisers were discussed. In both schemes the stability of the system is guaranteed, because the design of the adaptive power system stabilisers is based on the Lyapunov synthesis method. This is the main advantage of the AFPSSs compared to the neural-network-based adaptive PSS, which was discussed in Chapter 7. Another advantage of the AFPSSs compared to the neural-network-based APSS is the possibility of choosing the initial FPSS based on some linguistic rules, which can be obtained from experts or derived by the designer. This fact makes the AFPSSs useful even before on-line adaptation.

Comparing the indirect AFPSS with the direct AFPSS, it is revealed that the direct scheme has some advantages over the indirect one. Simulation results shows that the direct AFPSS damps the system oscillations more effectively. Moreover, the structure of the direct AFPSS is simpler and its design is straightforward. This makes its implementation easier. Also the computation time for the direct AFPSS is less than the indirect AFPSS and this makes its implementation feasible.

Chapter 9

Comparing Performances of the Proposed PSSs

9.1 Overview of the Chapter

This chapter compares the performances of the proposed power system stabilisers described in the previous chapters.

9.2 Introduction

In this chapter responses of the power system stabilisers designed in the previous chapters will be obtained using nonlinear simulations. Since the main objective of the thesis is to design various PSSs based on fuzzy logic and neural networks for an industrial cogenerator and since the power system connected to the cogenerator can be considered as an infinite bus, the simulation studies are performed using a one-machine infinite-bus model.

The proposed PSSs will be compared in various operating conditions. Although the plots of various quantities of the power system such as electric output power, reactive power, terminal voltage, and output voltage of the excitation system have been obtained in simulation studies, for brevity only the torque angle response, the

speed deviation response and the stabilising signal will be shown.

Advantages and disadvantages of the PSSs will be discussed at the end. Since the performances of the two fixed parameters FPSS (namely the Mamdani-Type and the polar FPSSs) are very similar, only the Mamdani-Type FPSS has been presented. Also, since the performance of the FPSS tuned by a fuzzy logic system is similar to the FPSS tuned by a neural network (the NFPSS), only the latter has been discussed.

The CPSS designed in Chapter 4 will be used as a bench-mark for comparisons. Also the performance index J_p defined by equation (6.5) (Section 6.3.3) is used to compare different results. The smaller the value of J_p, the better the performance of the PSS.

The machine parameters, provided by the manufacturer, and the parameters used for the excitation system are given in Appendix A. The machine parameters are converted to the fundamental parameters as explained in reference [1], so that they suit the pu system of the mathematical model. Voltage at the infinite bus (E_B) is considered to be 1.0 pu. The reactance of the transformer is (X_{tr}) is 0.05 pu.

Three sets of operating conditions are considered for simulation studies as follows. In all three cases P_o and the specified power factor are at the machine terminal.

Case 1) Normal conditions, i.e., $P_o = 0.9$ pu with a power factor of 0.9 lagging with the transmission line parameters $R_{e1} = 0.06$ pu, $R_{e2} = 0.06$ pu, $X_{e1} = 0.3$ pu and $X_{e2} = 0.3$ pu.

Case 2) Leading power factor operating point, i.e., $P_o = 0.85$ pu with a power factor

of 0.9 leading with the transmission line parameters R_{e1} = 0.06 pu, R_{e2} = 0.06 pu, X_{e1} = 0.3 pu and X_{e2} = 0.3 pu.

Case 3) Heavy reactive load with weak connections, i.e., P_o = 0.7 pu with a power factor of 0.7 lagging with the transmission line parameters R_{e1} = 0.1 pu, R_{e2} = 0.1 pu, X_{e1} = 0.9 pu and X_{e2} = 0.9 pu.

9.3 Comparing the Fixed-Parameter FPSS with the CPSS

9.3.1 Step change in the reference voltage

a) While the power system was operating as in Case 1, a disturbance of 0.1 pu decrease in reference voltage was applied at t = 0.1 seconds. The reference voltage was returned to the original value after 5 seconds. The system torque angle response for both CPSS and the fixed-parameters FPSS is shown in Fig. 9.1. Also, the response of the system without any PSS is shown in the figure. The performance

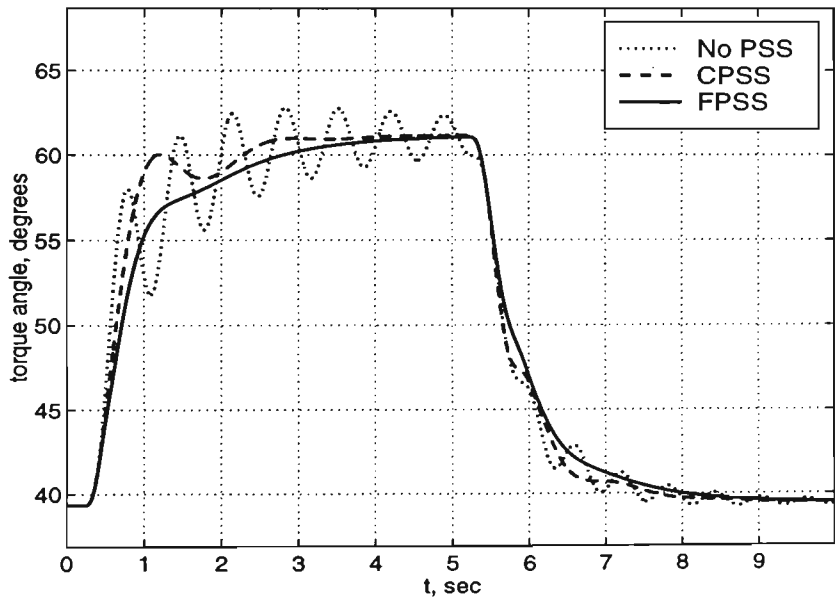


Fig. 9.1 The torque angle responses of CPSS and FPSS for a disturbance in the reference voltage (Case 1)

index for the system with no PSS is $J_p = 1282$, for the system with CPSS is $J_p = 521$, and for the system with FPSS is $J_p = 494$. Fig. 9.2 shows the terminal

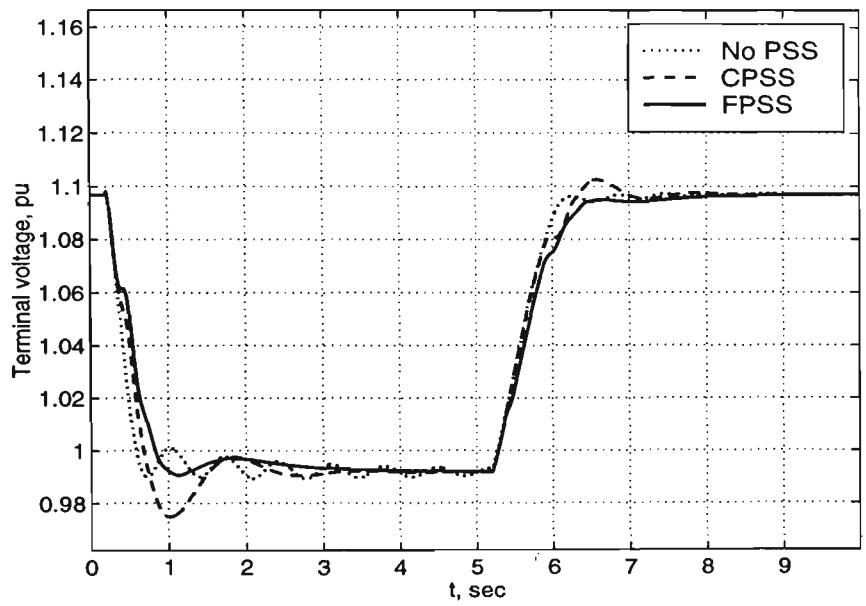


Fig. 9.2 The terminal voltage responses of CPSS and FPSS for a disturbance in the reference voltage (Case 1)

voltage of the system in this case. Since both the CPSS and the FPSS are designed for this operating condition, their performance are compatible in this case.

b) While the system was operating as in Case 2, the same disturbance was applied. The system torque angle response is shown in Fig. 9.3. One may wonder how the torque angle can be more than 90 degrees and the machine still remains in synchronism. The answer is that one should differentiate between the internal torque angle of the machine (δ_i), which is defined as the angle between q-axis of the machine and its terminal voltage, and the torque angle with respect to the common reference of the system (δ), which is defined as the angle between the q-axis of the machine and the infinite bus voltage. The torque angle referred to in the simulation studies is the torque angle (δ) with respect to the common reference of the system

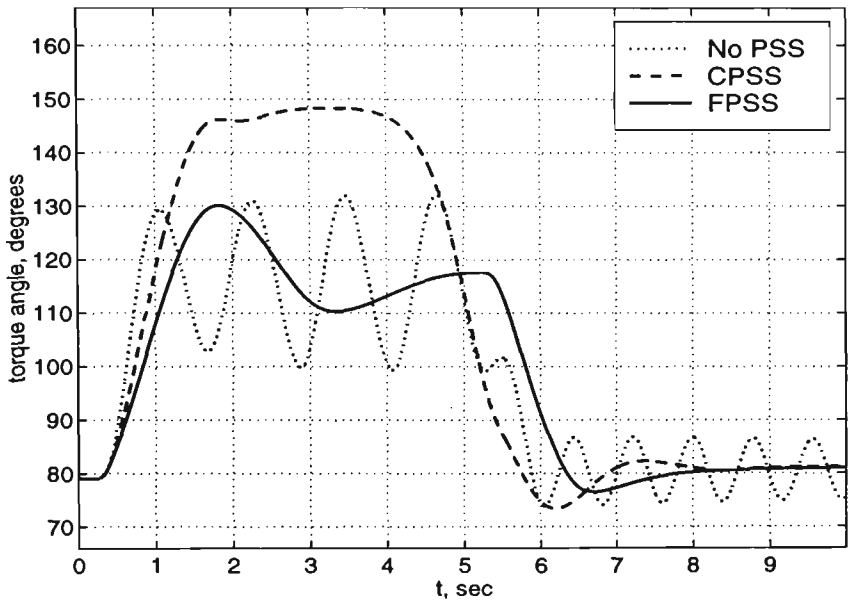


Fig. 9.3 The torque angle responses of CPSS and FPSS for a disturbance in the reference voltage (Case 2)

$(\tilde{E}_B)$. Fig. 9.4 illustrates these definitions.

The performance index for the system with no PSS is $J_p = 6570$, for the system with CPSS is $J_p = 1837$, and for the system with FPSS is $J_p = 1327$. Fig.

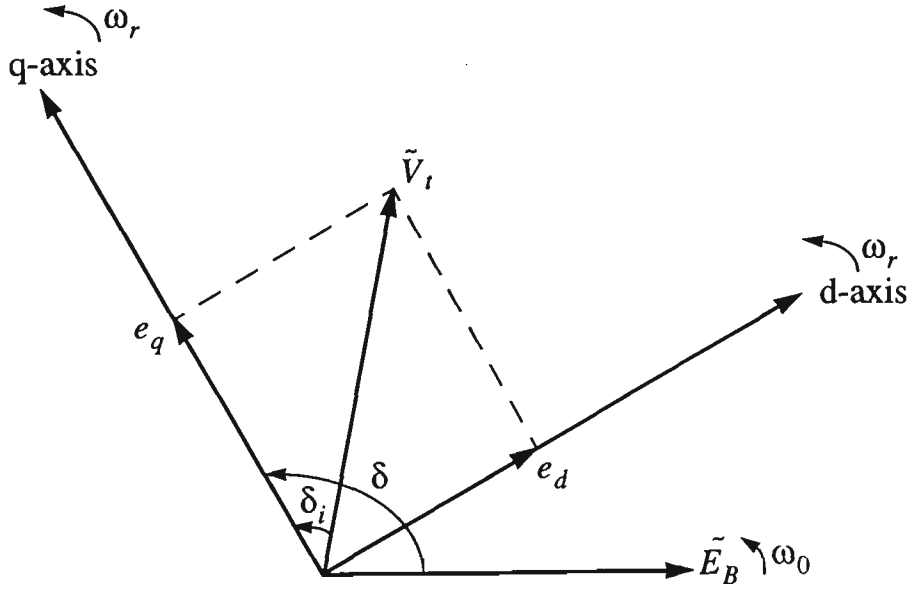


Fig. 9.4 Phasor diagram showing the torque angle δ referred to in the simulation studies

9.5 shows the terminal voltage responses of the system. It is observed from the

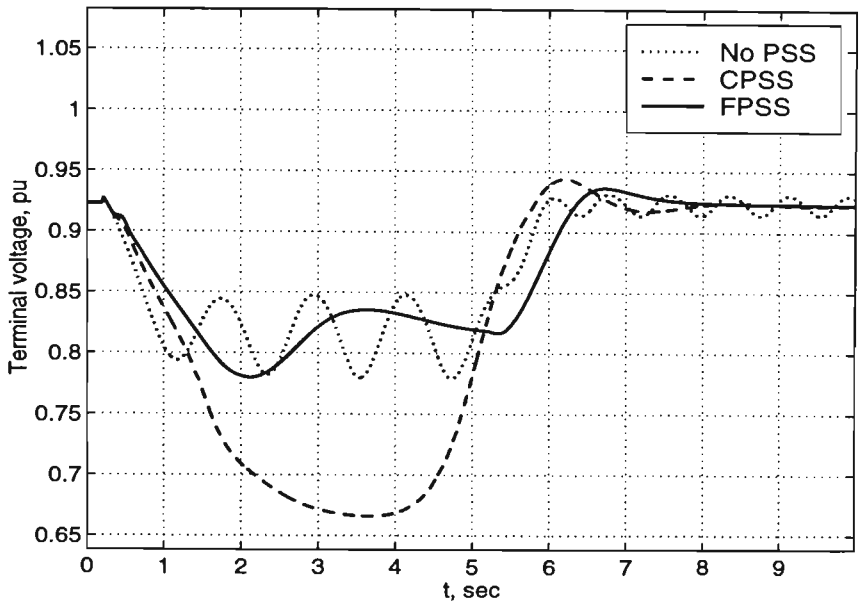


Fig. 9.5 The terminal voltage responses of CPSS and FPSS for a disturbance in the reference voltage (Case 2)

figures that in this case the system with the CPSS is on the edge of instability. On the other hand, the FPSS has a better performance, though there is some room for improvement.

c) While the system was operating as in Case 3, the same disturbance as in the previous cases was applied. The system torque angle response is shown in Fig. 9.6 and the terminal voltage of the system is shown in Fig. 9.7. In this case the response of the system with the CPSS is more oscillatory than the response of the system with the FPSS.

9.3.2 Step change in the mechanical input power

a) A disturbance of 0.1 pu increase in the input mechanical power was applied to the system at $t = 0.1$ seconds, while the system operating conditions were as in Case 1. The disturbance was removed after 5 seconds and the system returned to the

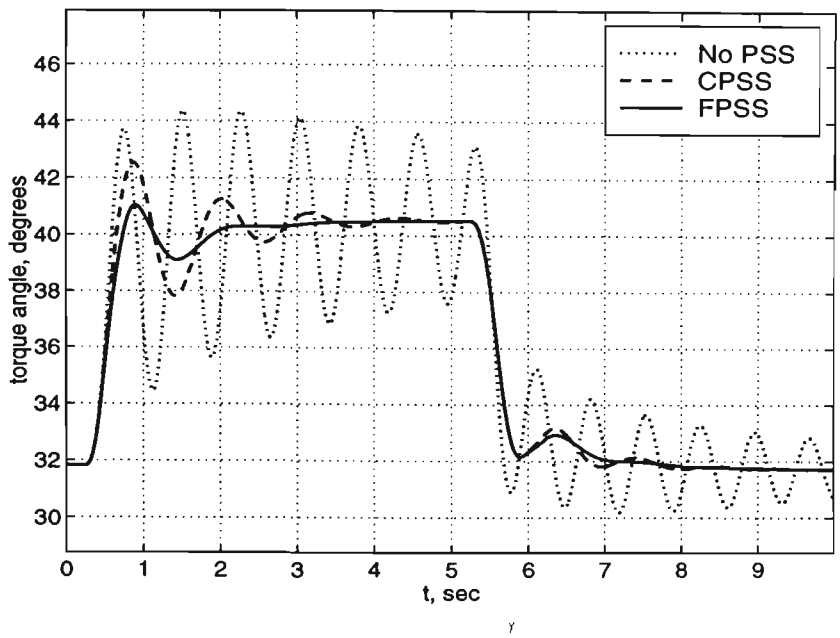


Fig. 9.6 The torque angle responses of CPSS and FPSS for a disturbance in the reference voltage (Case 3)

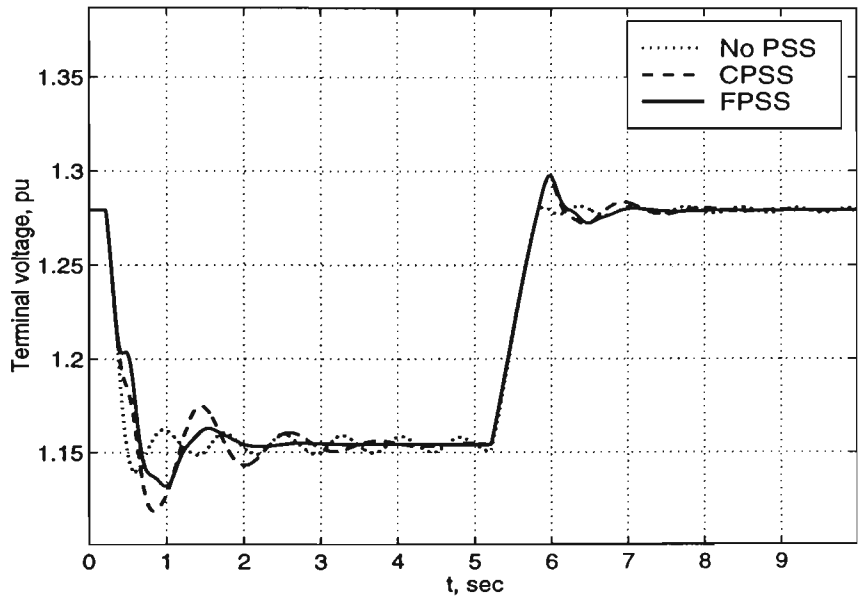


Fig. 9.7 The terminal voltage responses of CPSS and FPSS for a disturbance in the reference voltage (Case 3)

original state. The system torque angle response is shown in Fig. 9.8 and the terminal voltage of the machine is shown in Fig. 9.9. The performance index for the system

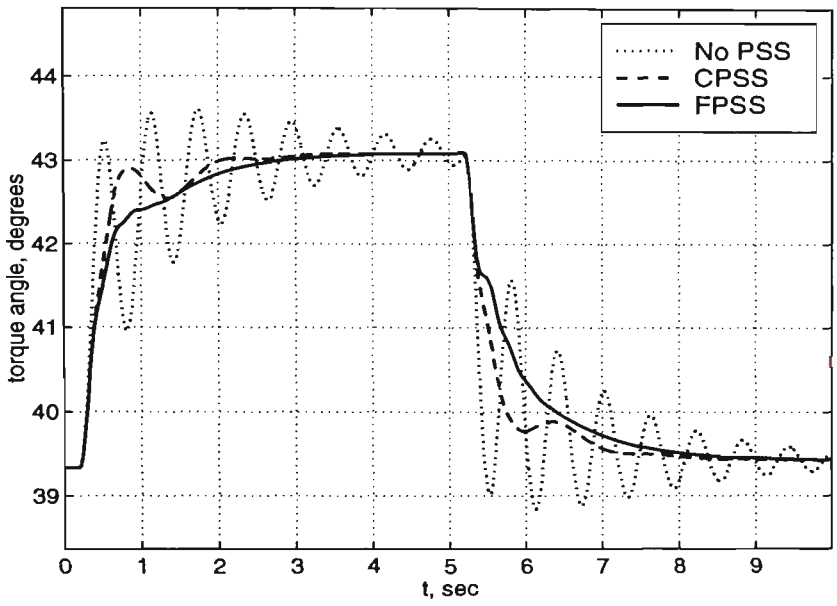


Fig. 9.8 The torque angle responses of CPSS and FPSS for a disturbance in the input mechanical power (Case 1)

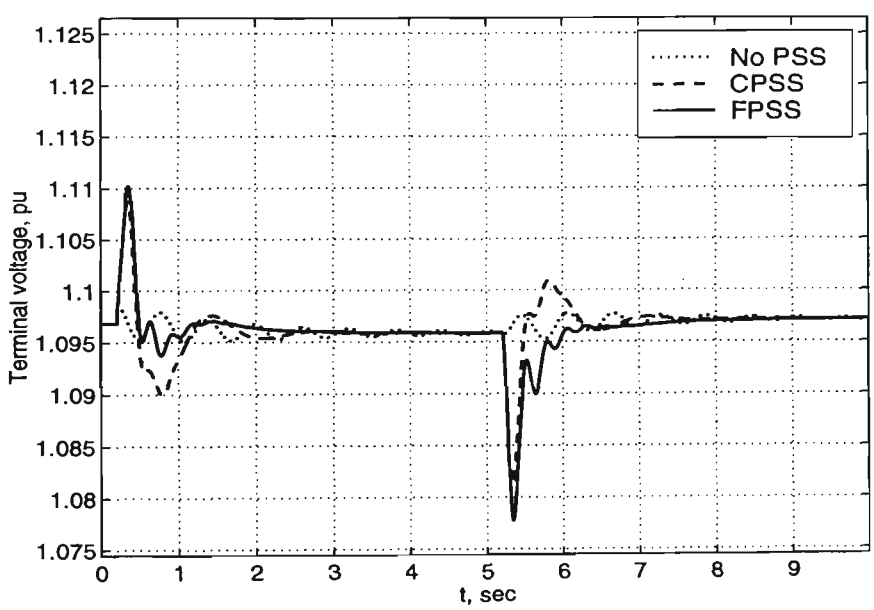


Fig. 9.9 The terminal voltage responses of CPSS and FPSS for a disturbance in the input mechanical power (Case 1)

with no PSS is $J_p = 599$, for the system with CPSS is $J_p = 87.2$, and for the system with FPSS is $J_p = 78.5$. As before, the responses are comparable in this case.

b) The same disturbance was applied to the system while the system operating conditions were as in Case 2. The system torque angle response is shown in Fig. 9.10

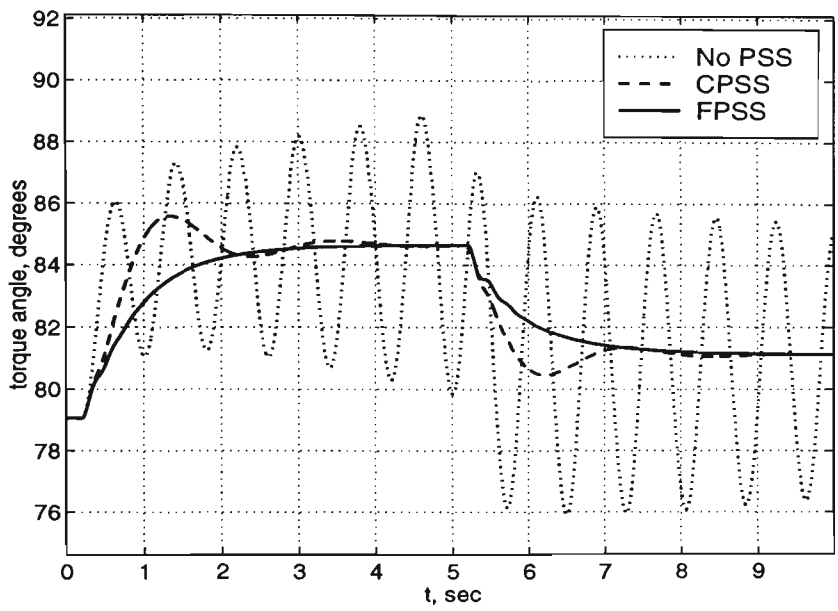


Fig. 9.10 The torque angle responses of CPSS and FPSS for a disturbance in the input mechanical power (Case 2)

and the terminal voltage is shown in Fig. 9.11. The performance index for the system with no PSS is $J_p = 4061$, for the system with CPSS is $J_p = 143$, and for the system with FPSS is $J_p = 84$.

c) The same disturbance was applied to the system while the system operating conditions were as in Case 3. The system torque angle response is shown in Fig. 9.12 and the terminal voltage is shown in Fig. 9.13. The performance index for the system with no PSS is $J_p = 841$, for the system with CPSS is $J_p = 158.6$, and for the system with FPSS is $J_p = 70.3$.

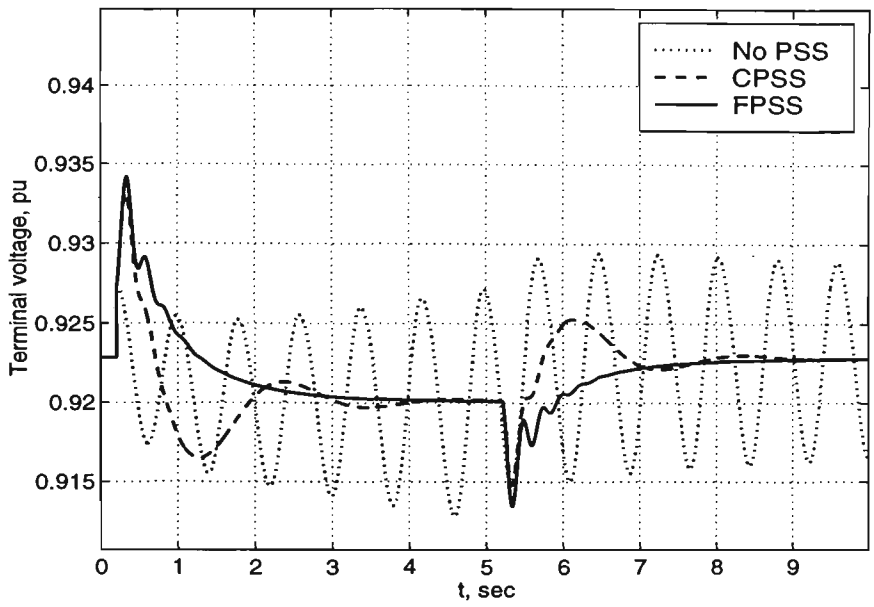


Fig. 9.11 The terminal voltage responses of CPSS and FPSS for a disturbance in the input mechanical power (Case 2)

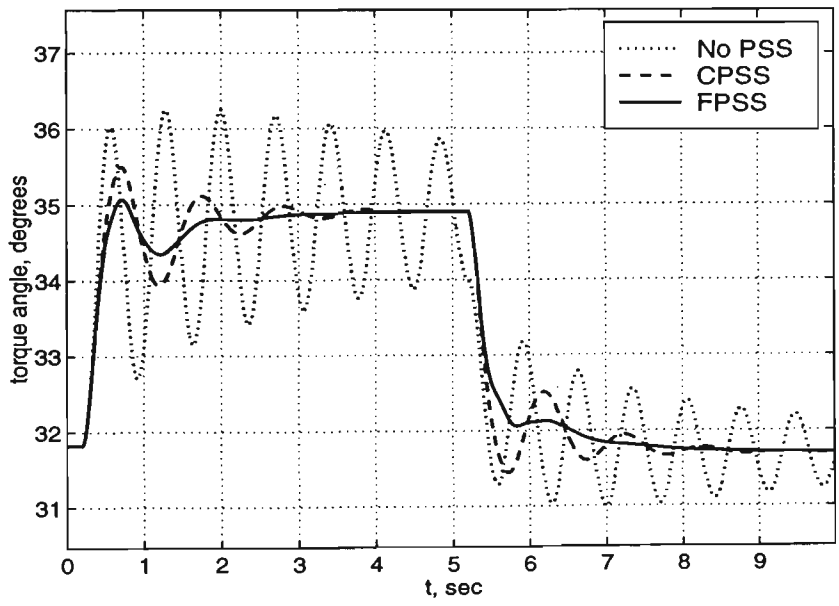


Fig. 9.12 The torque angle responses of CPSS and FPSS for a disturbance in the input mechanical power (Case 3)

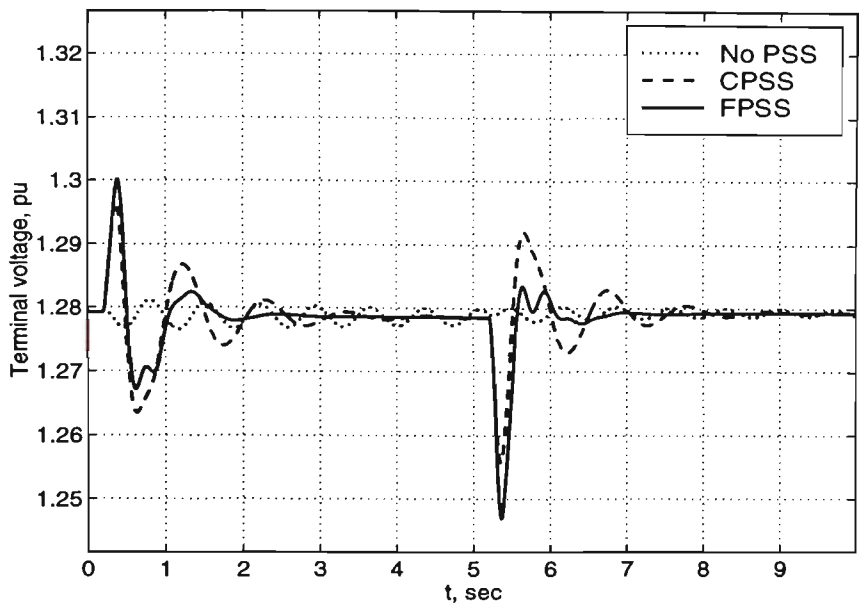


Fig. 9.13 The terminal voltage responses of CPSS and FPSS for a disturbance in the input mechanical power (Case 3)

9.3.3 Three-Phase to ground fault test

9.3.3.1 Normal load and switching off one line

While the system was operating as in Case 1, a three-phase to ground fault occurred at one end of one of the transmission lines close to the bus-bar connected to the transformer near the generator (refer to Fig. 3.7). The faulty line was switched off after 40 msec. The system torque angle response is shown in Fig. 9.14. Also, the terminal voltage of the generator is shown in Fig. 9.15 and the stabilising signals are shown in Fig. 9.16.The performance index for the system with no PSS is $J_p = 1903$, for the system with CPSS is $J_p = 92.3$, and for the system with FPSS is $J_p = 25.5$. Obviously, the FPSS has a better performance.

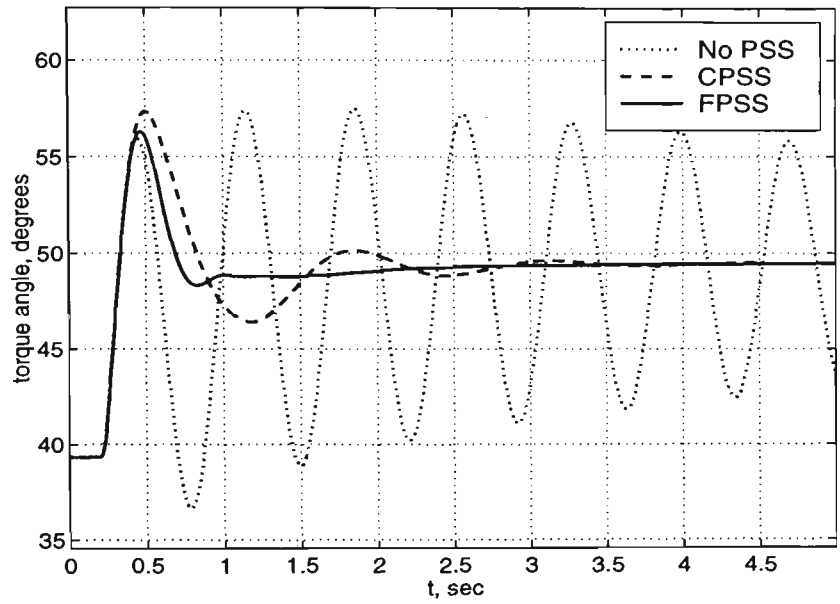


Fig. 9.14 The torque angle responses of CPSS and FPSS for a three-phase to ground fault (Case 1)

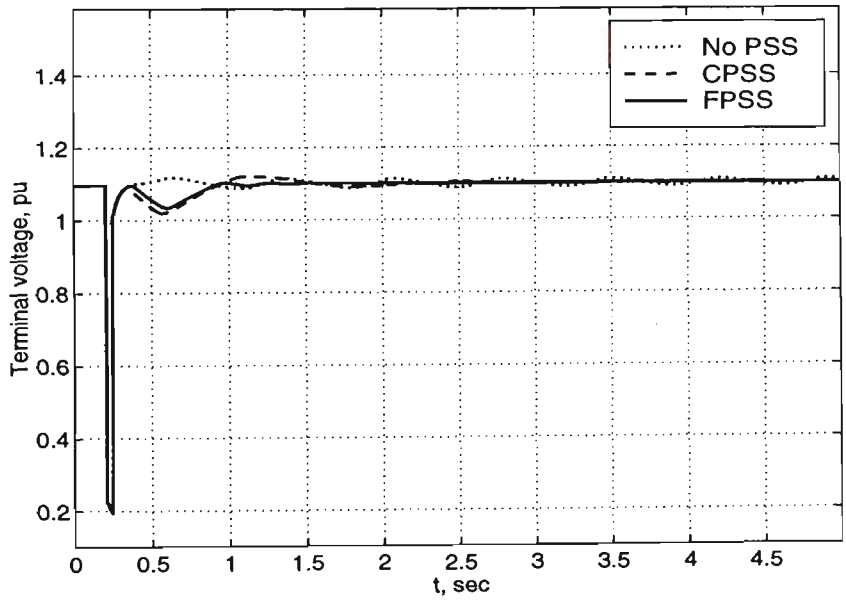


Fig. 9.15 The terminal voltage responses of CPSS and FPSS for a three-phase to ground fault (Case 1)

9.3.3.2 Leading power factor load and switching off one line

While the system was operating as in Case 2, the same fault occurred at the same position. The system torque angle response is shown in Fig. 9.17. The terminal

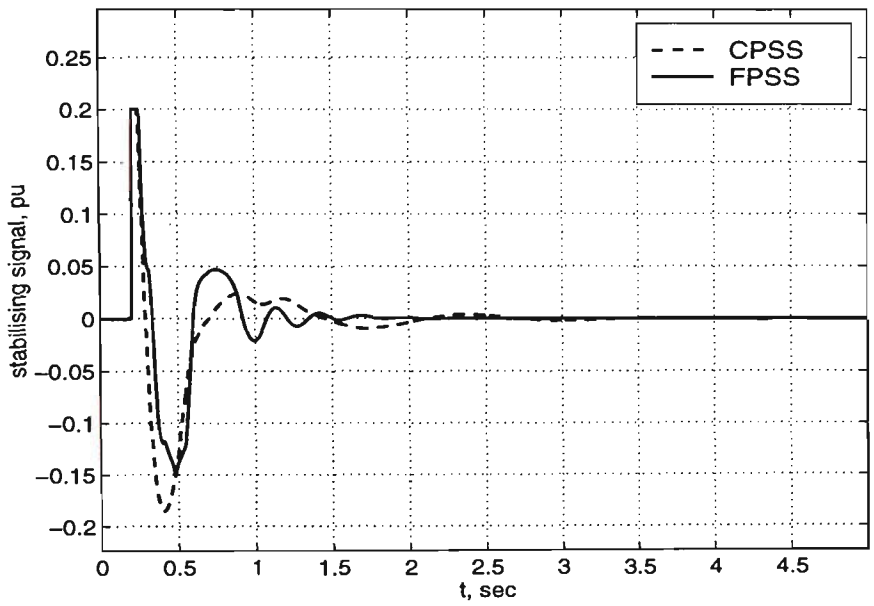


Fig. 9.16 The outputs of CPSS and FPSS for a three-phase to ground fault (Case 1)

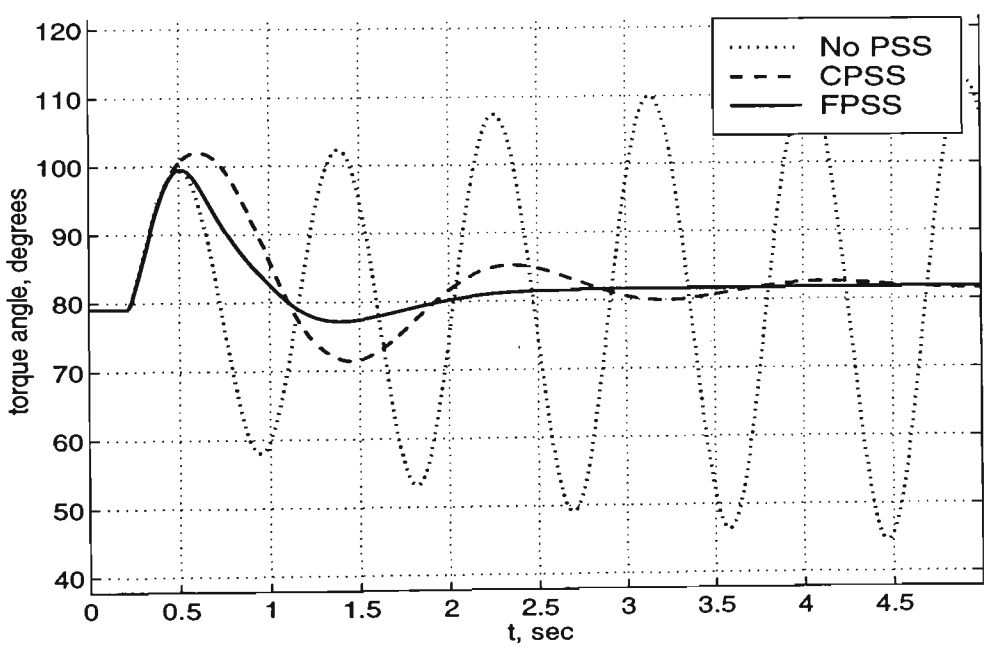


Fig. 9.17 The torque angle responses of CPSS and FPSS for a three-phase to ground fault (Case 2)

voltage of the generator is shown in Fig. 9.18 and the stabilisers output are shown in Fig. 9.19. The performance index for the system with no PSS is $J_p = 5897$, for the

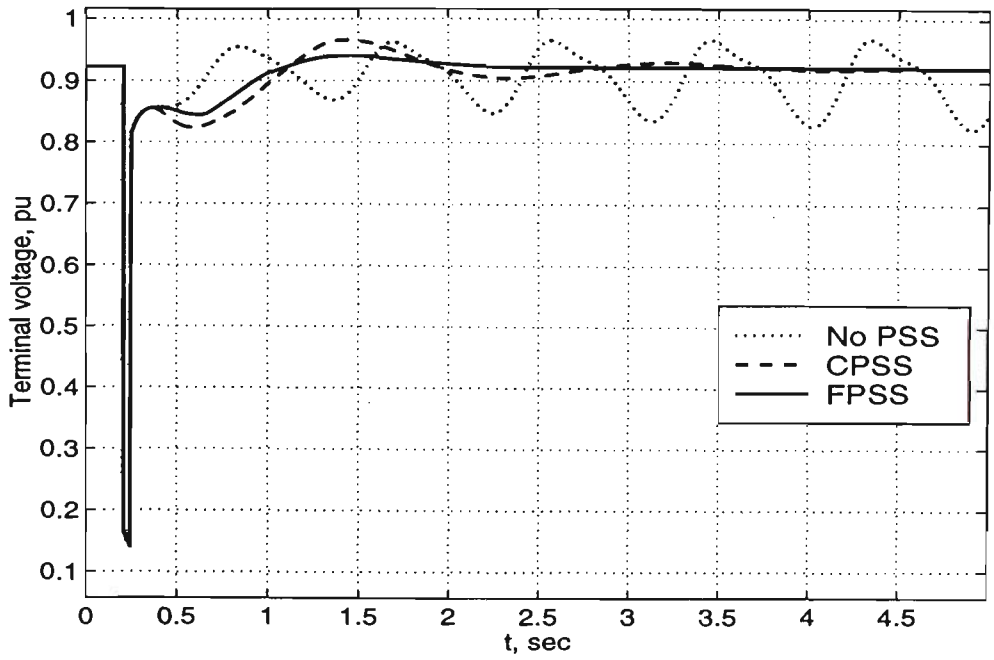


Fig. 9.18 The terminal voltage responses of CPSS and FPSS for a three-phase to ground fault (Case 2)

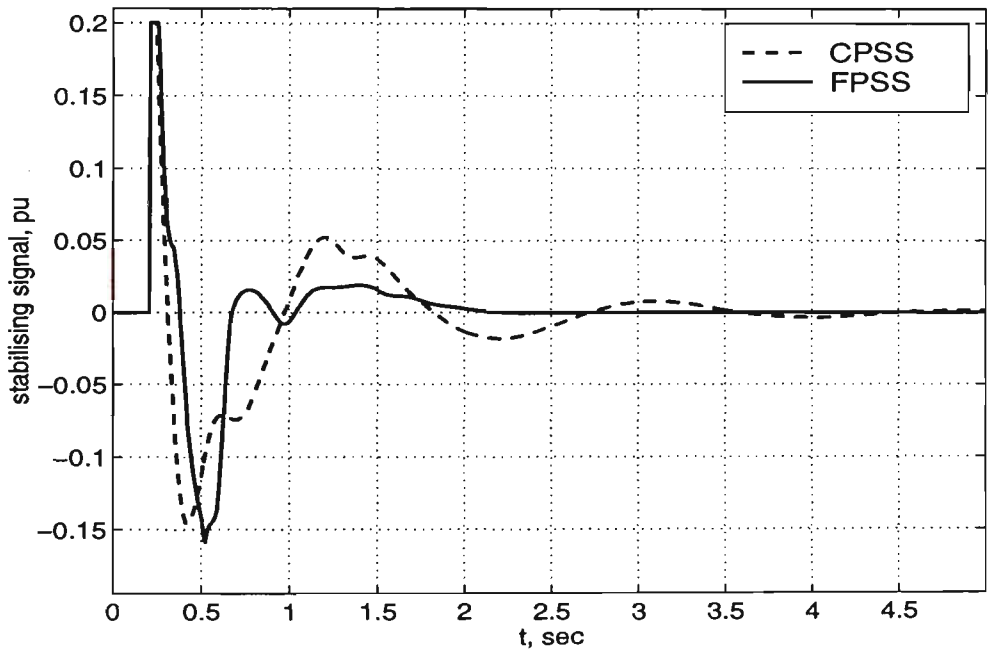


Fig. 9.19 The outputs of CPSS and FPSS for a three-phase to ground fault (Case 2)

system with CPSS is $J_p = 271.5$, and for the system with FPSS is $J_p = 87.9$.

9.3.3.3 Heavy reactive load and switching off one line

While the system was operating as in Case 3, the same fault occurred at the same position. The system torque angle response is shown in Fig. 9.20. The terminal

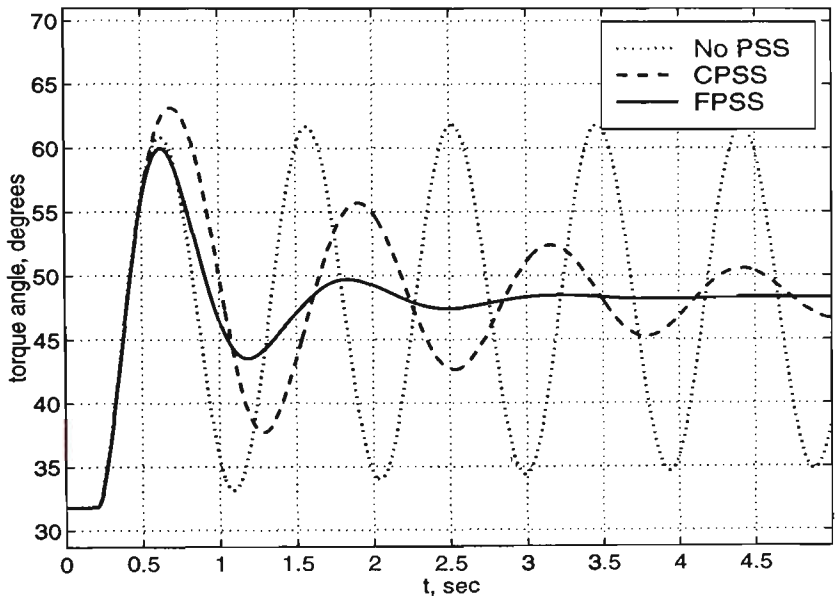


Fig. 9.20 The torque angle responses of CPSS and FPSS for a three-phase to ground fault and switching off the faulty line (Case 3)

voltage of the generator is shown in Fig. 9.21 and the stabilising control is shown in Fig. 9.22. The performance index for the system with no PSS is $J_p = 2478$, for the system with CPSS is $J_p = 655$, and for the system with FPSS is $J_p = 115.4$.

9.3.3.4 Normal load without disconnecting the faulty line

While the system was operating as in Case 1, a three-phase to ground fault occurred at the same position. The fault was cleared after 40 msec without disconnecting the faulty line. The system torque angle response is shown in Fig. 9.23. Also, the terminal voltage of the generator is shown in Fig. 9.24 and the stabilising control is shown in Fig. 9.25. The performance index for the system with no PSS is $J_p = 681.9$, for the system with CPSS is $J_p = 48.4$, and for the system with FPSS

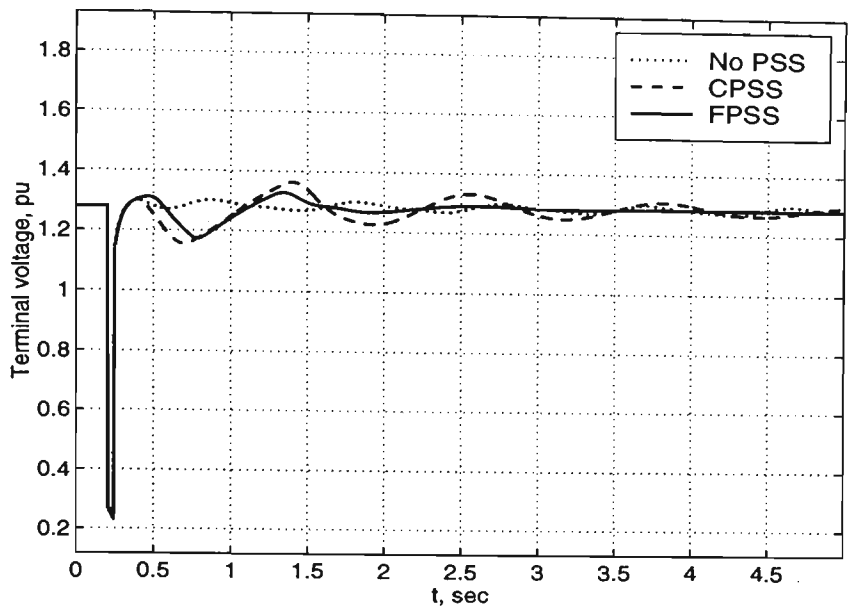


Fig. 9.21 The terminal voltage responses of CPSS and FPSS for a three-phase to ground fault and switching off the faulty line (Case 3)

is $J_p = 31.6$.

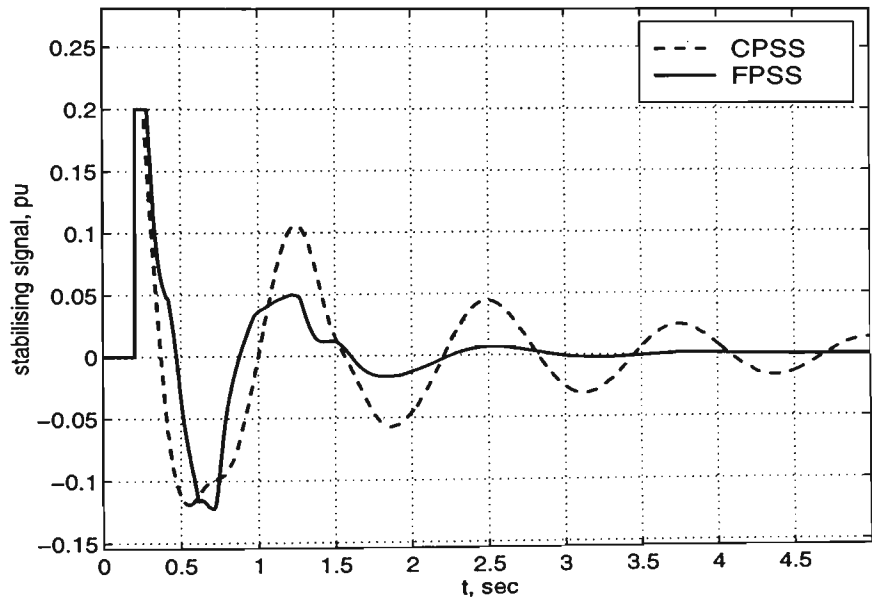


Fig. 9.22 The outputs of CPSS and FPSS for a three-phase to ground fault and switching off the faulty line (Case 3)

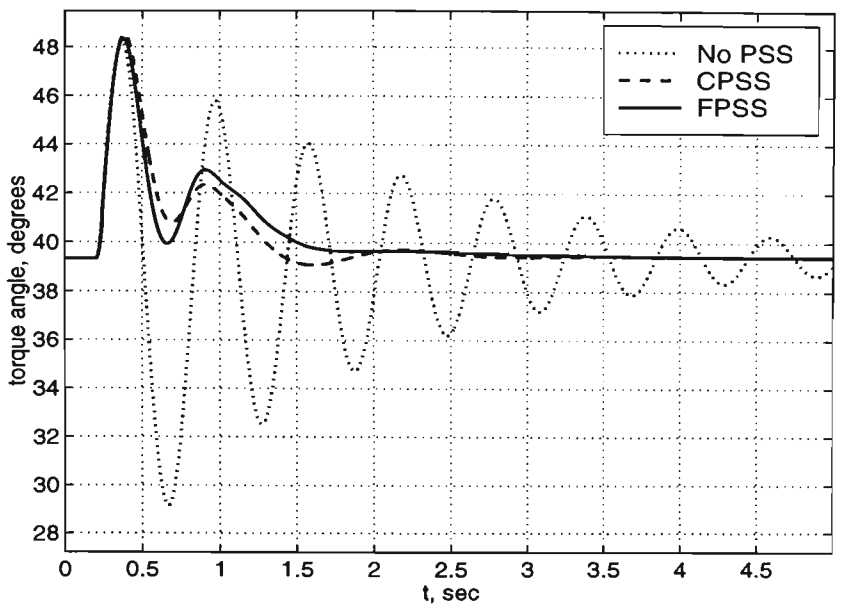


Fig. 9.23 The torque angle responses of CPSS and FPSS for a three-phase to ground fault with the faulty line remaining after fault clearance (Case 1)

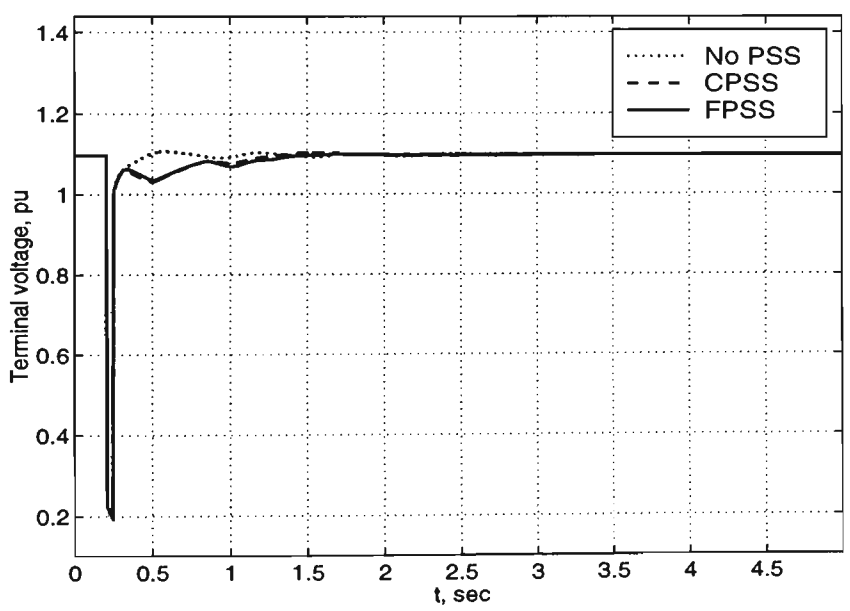


Fig. 9.24 The terminal voltage responses of CPSS and FPSS for a three-phase to ground fault with the faulty line remaining after fault clearance (Case 1)

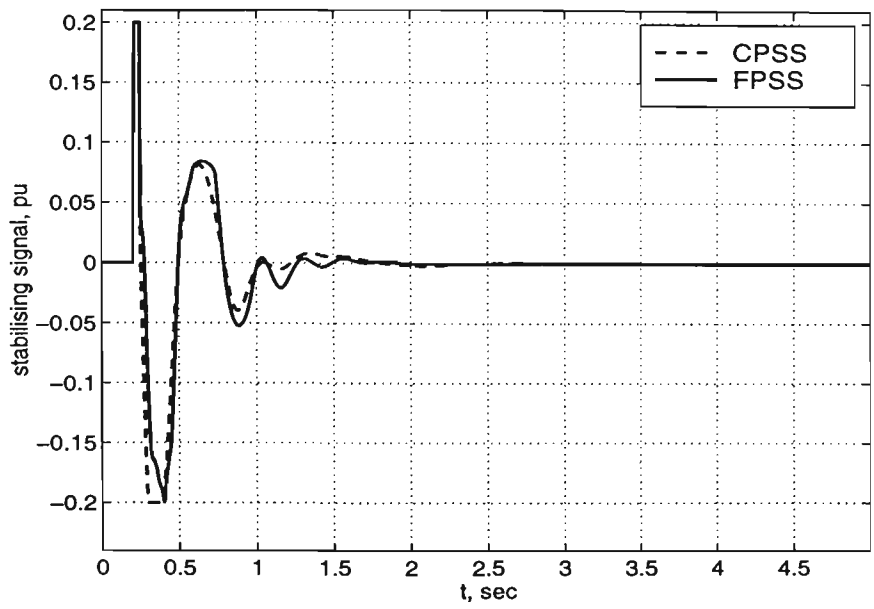


Fig. 9.25 The outputs of CPSS and FPSS for a three-phase to ground fault with the faulty line remaining after fault clearance (Case 1)

9.3.3.5 Leading power factor without disconnecting the faulty line

While the system was operating as in Case 2, the same fault as in the previous sub-section occurred at the same position. The system torque angle response is shown in Fig. 9.26. Also, the terminal voltage of the generator is shown in Fig. 9.27 and the stabilisers outputs are shown in Fig. 9.28. The performance index for the system with no PSS is $J_p = 3417$, for the system with CPSS is $J_p = 105.5$, and for the system with FPSS is $J_p = 42.3$.

9.3.3.6 Heavy reactive load without disconnecting the faulty line

For Case 3, the same fault as in the previous sub-section occurred. The system torque angle response is shown in Fig. 9.29. Also, the terminal voltage of the generator is shown in Fig. 9.30 and the stabilisers outputs are shown in Fig. 9.31. The performance index for the system with no PSS is $J_p = 1239$, for the system with

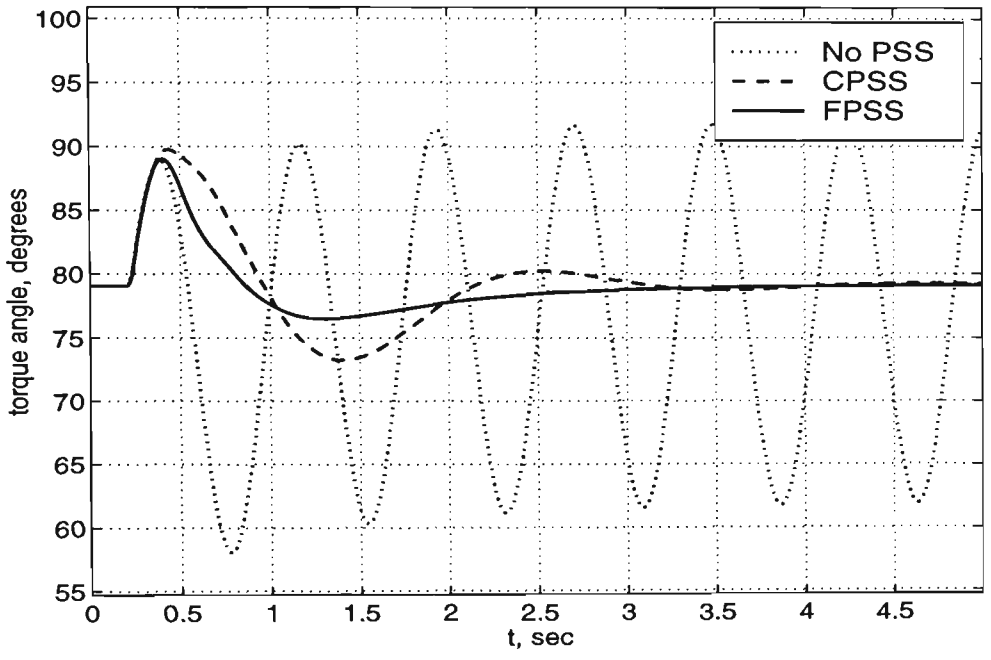


Fig. 9.26 The torque angle responses of CPSS and FPSS for a three-phase to ground fault with the faulty line remaining after fault clearance (Case 2)

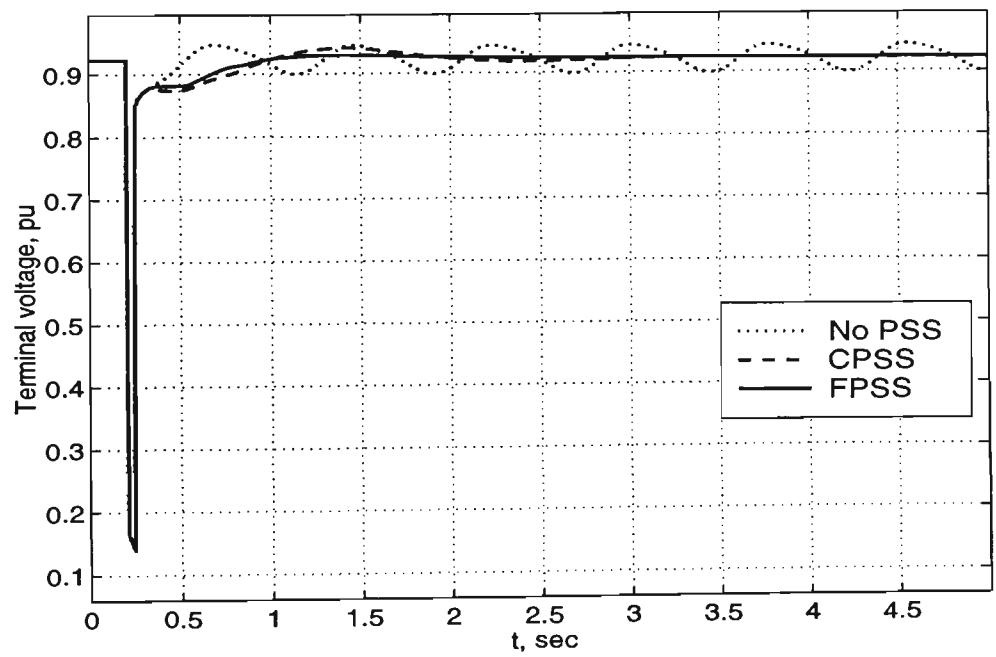


Fig. 9.27 The terminal voltage responses of CPSS and FPSS for a three-phase to ground fault with the faulty line remaining after fault clearance (Case 2)

CPSS is $J_p = 82.4$, and for the system with FPSS is $J_p = 47.8$.

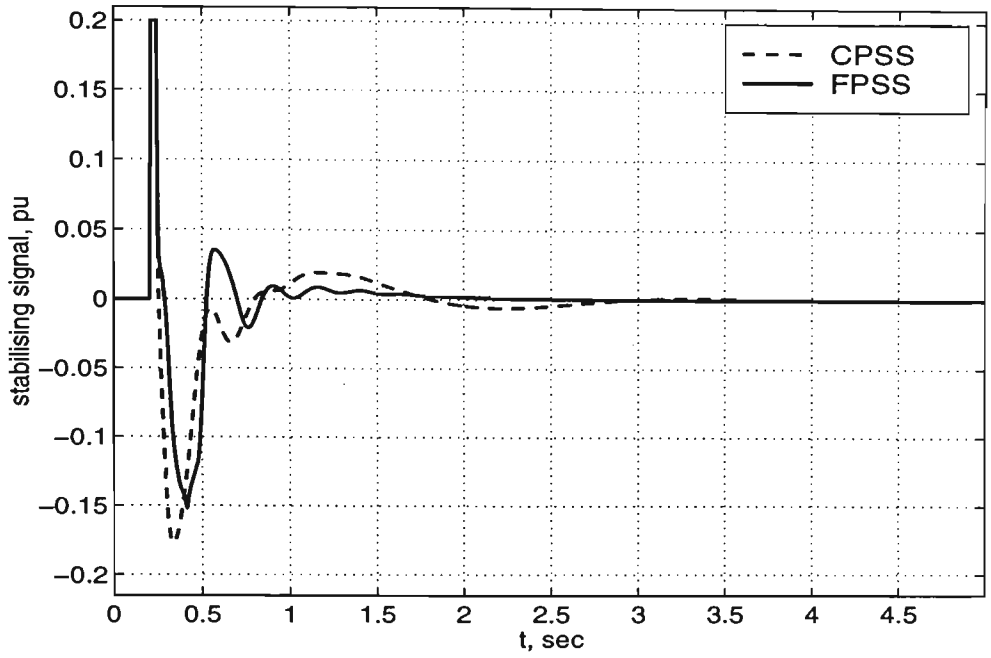


Fig. 9.28 The outputs of CPSS and FPSS for a three-phase to ground fault with the faulty line remaining after fault clearance (Case 2)

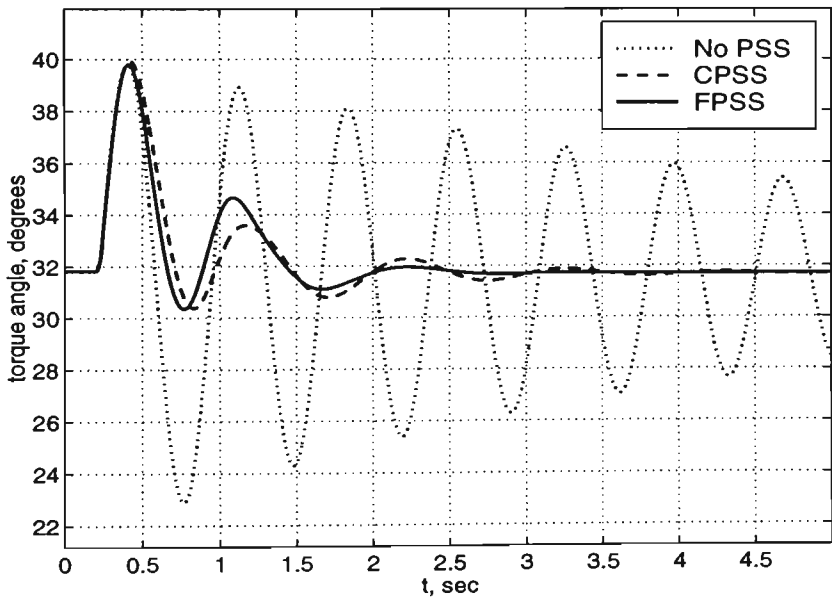


Fig. 9.29 The torque angle responses of CPSS and FPSS for a three-phase to ground fault with the faulty line remaining after fault clearance (Case 3)

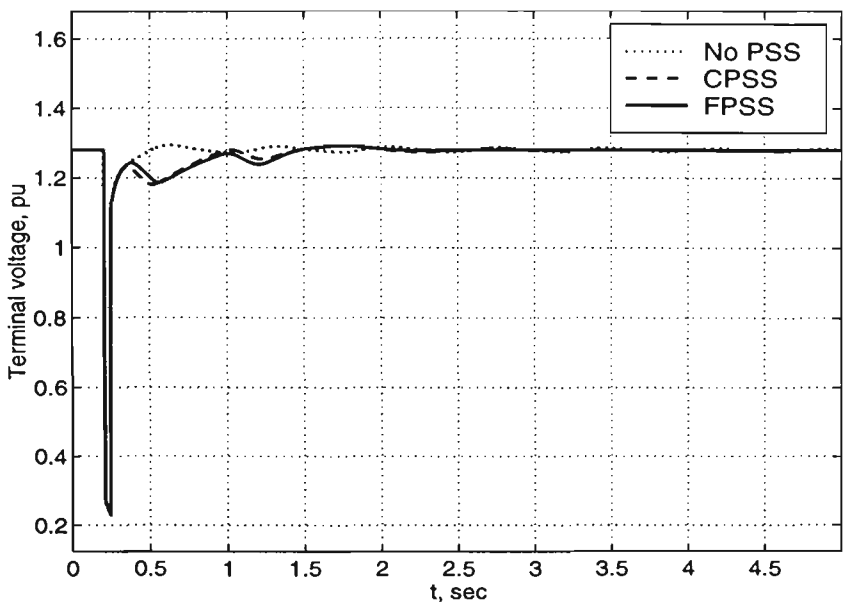


Fig. 9.30 The terminal voltage responses of CPSS and FPSS for a three-phase to ground fault with the faulty line remaining after fault clearance (Case 3)

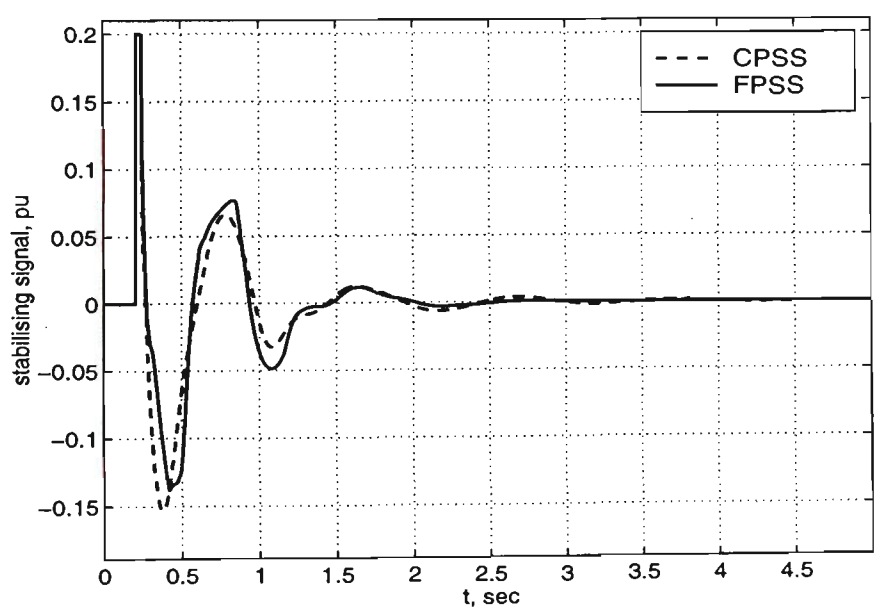


Fig. 9.31 The outputs of CPSS and FPSS for a three-phase to ground fault with the faulty line remaining after fault clearance (Case 3)

9.3.4 Different oscillation-mode test

The main purpose of the PSSs is to damp out the low frequency oscillations of the power system under different disturbances. The CPSS normally needs to be redesigned for each power system application and the FPSS has to be re-tuned if the system configuration changes. However, the FPSS has the capability of accepting imprecision and even vagueness in the system parameters to some extent.

In this test different machine inertia was used to introduce different oscillation modes. Tests were conducted for machine per-unit inertia, H, changing from 2.0 to 5.0 seconds (the nominal value is 3.8 seconds). The machine was working under normal conditions (Case 1) while a three-phase to ground occurred at the sending end of the transmission line. The fault was cleared after 40 msec by disconnecting the faulty line. For per unit inertia being $H = 2$ seconds, system torque angle responses for the system without any PSS, with the CPSS and with the FPSS are shown in Fig. 9.32 and the stabilisers output are shown in Fig. 9.33. The performance index is

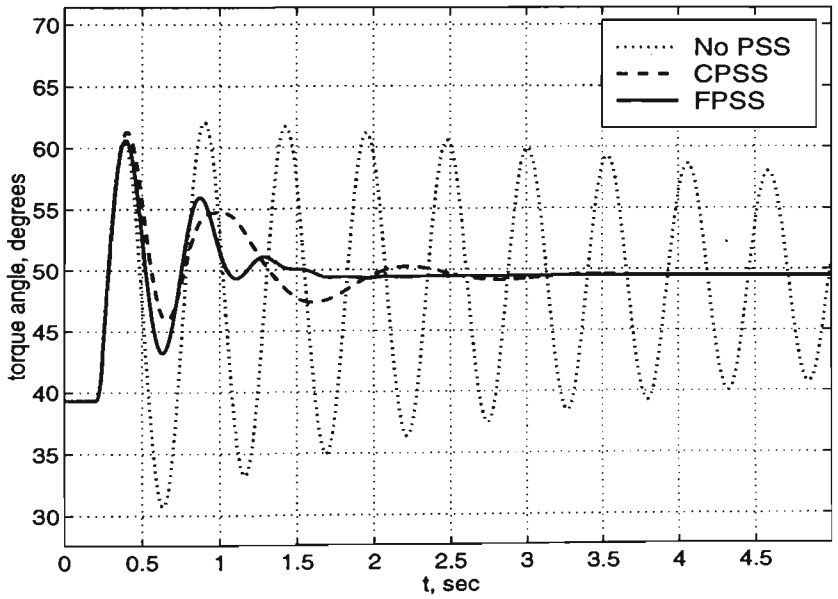


Fig. 9.32 The torque angle responses of CPSS and FPSS for a three-phase to ground fault with H = 2 (Case 1)

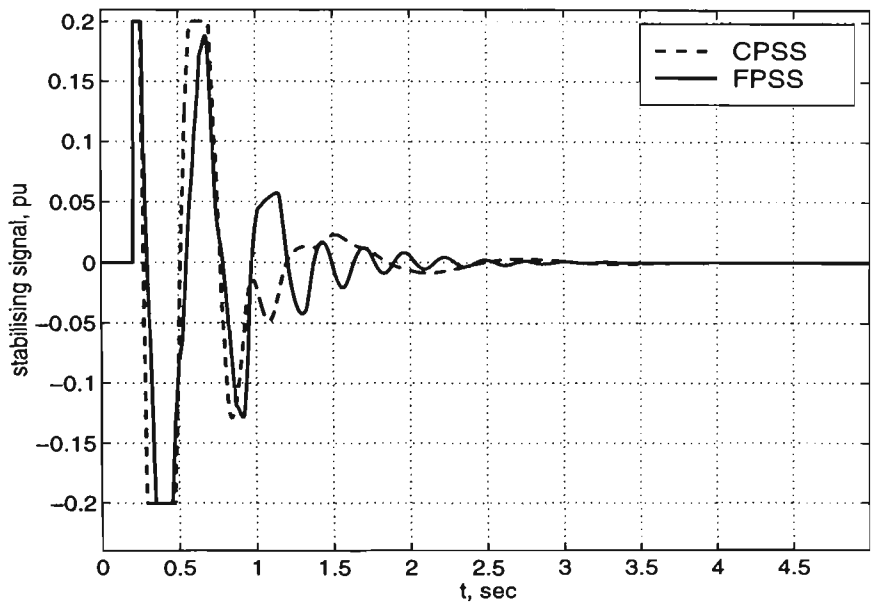


Fig. 9.33 The outputs of CPSS and FPSS for a three-phase to ground fault with H = 2 (Case 1)

J_p = 3541 for the system with no PSS, J_p = 145 for the system with CPSS, and J_p = 85.4 for the system with FPSS.

For H = 5 seconds, the torque angle responses and the stabilisers outputs are shown in Fig. 9.34 and Fig. 9.35,respectively. In this case the performance index is J_p = 1182 for the system with no PSS, J_p = 150 for the system with CPSS, and J_p = 37 for the system with FPSS.

9.4 Comparing the NFPSS with the CPSS

The performance of the NFPSS compared with the FPSS and CPSS is examined by conducting the three-phase to ground fault test introduced in sub-section 9.3.3 for three cases introduced in Section 9.2.

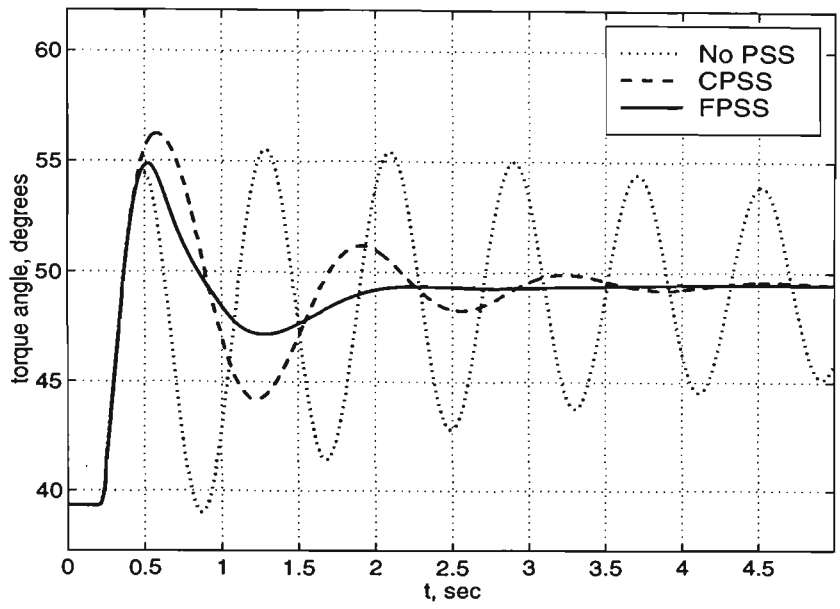


Fig. 9.34 The torque angle responses of CPSS and FPSS for a three-phase to ground fault with H = 5 (Case 1)

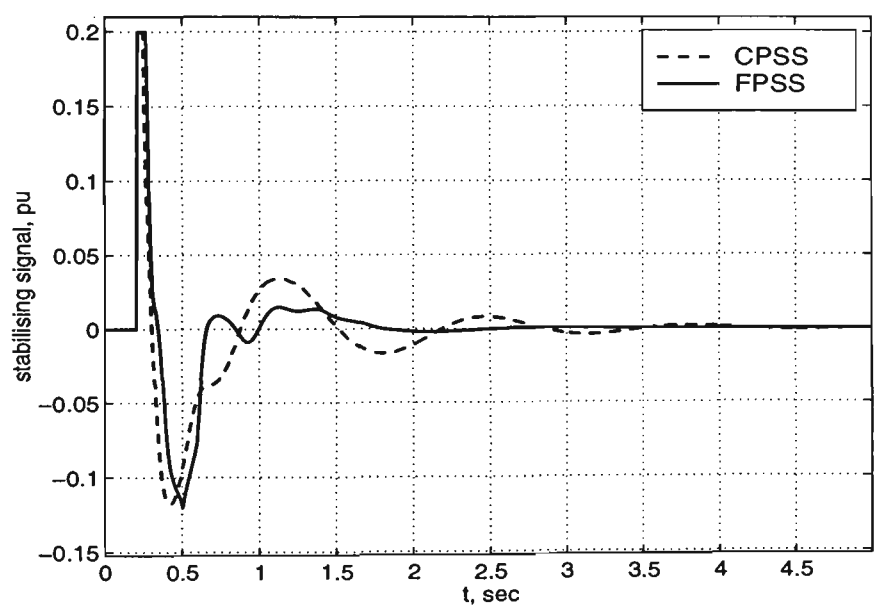


Fig. 9.35 The outputs of CPSS and FPSS for a three-phase to ground fault with H = 5 (Case 1)

9.4.1 Normal load and switching off one line

While the system was operating as in Case 1, a three-phase to ground fault occurred at the sending end of the transmission line. The faulty line was switched off after 40 msec. The system torque angle and speed deviation are shown in Fig. 9.36

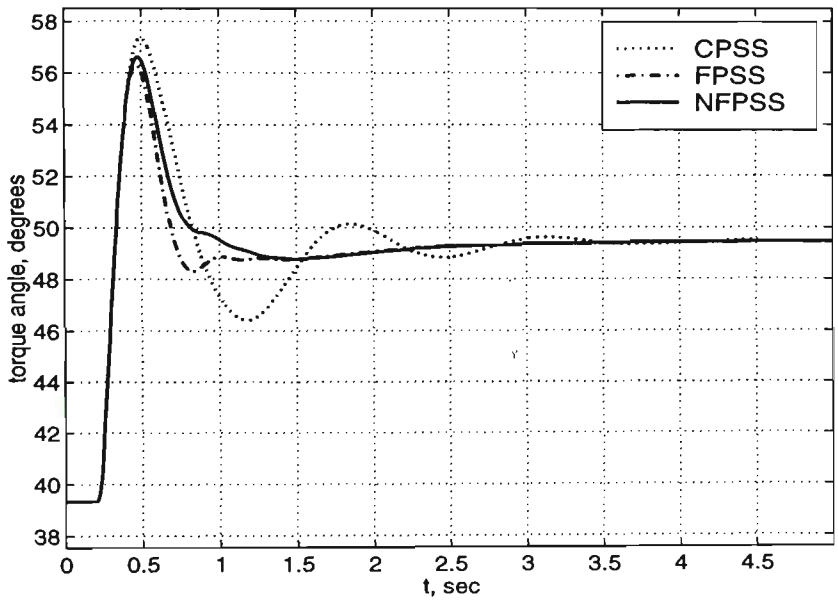


Fig. 9.36 The torque angle response of the NFPSS compared with the CPSS and FPSS for a three-phase to ground fault and switching off the faulty line (Case 1)

and Fig. 9.37, respectively. The performance index is $J_p = 92.3$ for the CPSS, $J_p = 25.5$ for the FPSS, and $J_p = 25.3$ for the NFPSS.

9.4.2 Leading power factor and switching off one line

While the system was operating as in Case 2, the same fault occurred at the sending end of the transmission line. The system torque angle and speed deviation are shown in Fig. 9.38 and Fig. 9.39, respectively. The performance index is $J_p = 271.5$ for the CPSS, $J_p = 87.9$ for the FPSS, and $J_p = 49.3$ for the NFPSS.

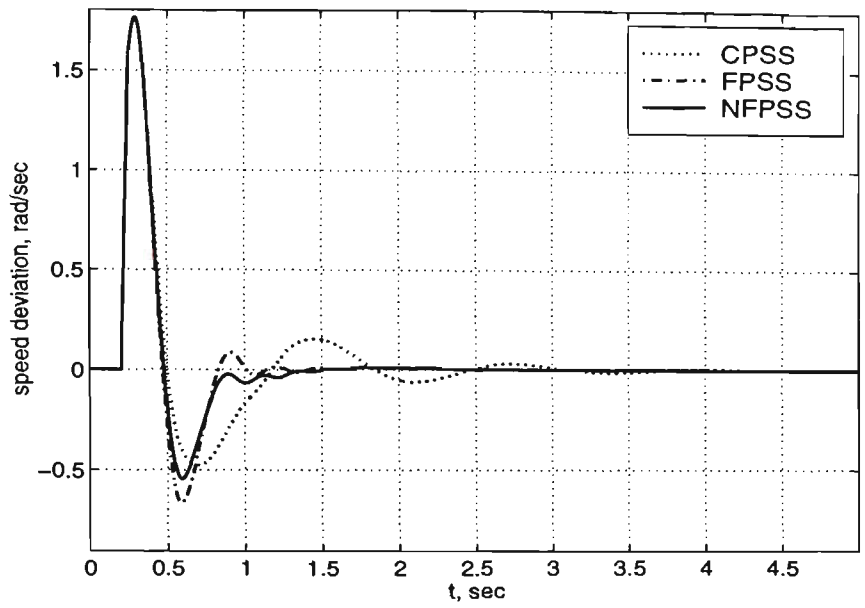


Fig. 9.37 The speed deviation response of the NFPSS compared with the CPSS and FPSS for a three-phase to ground fault and switching off the faulty line (Case 1)

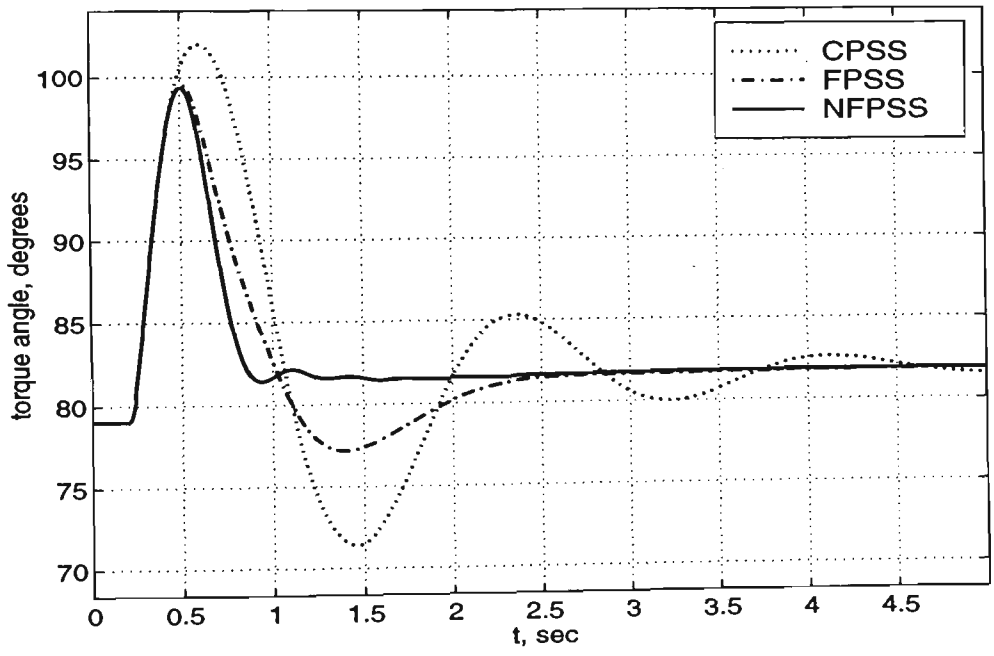


Fig. 9.38 The torque angle response of the NFPSS compared with the CPSS and FPSS for a three-phase to ground fault and switching off the faulty line (Case 2)

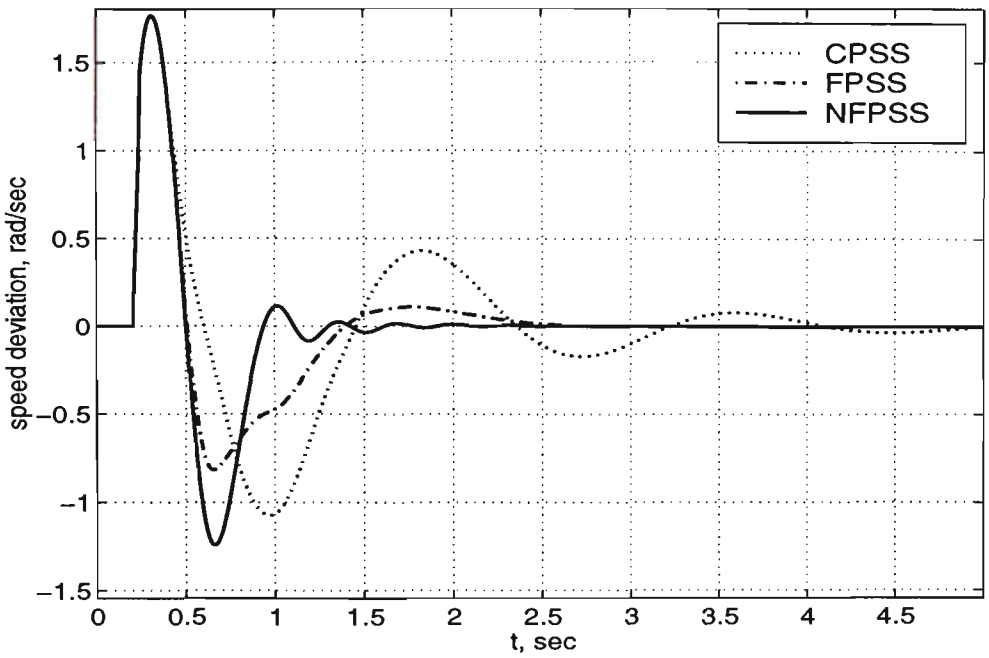


Fig. 9.39 The speed deviation response of the NFPSS compared with the CPSS and FPSS for a three-phase to ground fault and switching off the faulty line (Case 2)

9.4.3 Heavy reactive load and switching off one line

While the system was operating as in Case 3, the system was disturbed by the same fault. The system torque angle and speed deviation are shown in Fig. 9.40 and Fig. 9.41, respectively. The performance index is J_p = 655 for the CPSS, J_p = 115.4 for the FPSS, and J_p = 55.8 for the NFPSS.

9.5 Comparing the AFPSSs with the FPSS and CPSS

The performance of the indirect and direct adaptive fuzzy power system stabilisers (AFPSSs) are compared with the CPSS by conducting the three-phase to ground fault test as in the previous section.

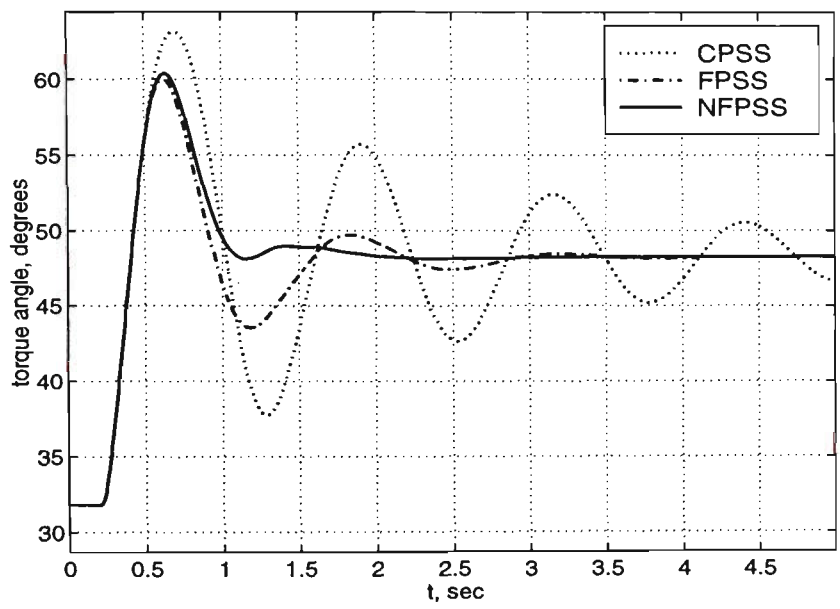


Fig. 9.40 The torque angle response of the NFPSS compared with the CPSS and FPSS for a three-phase to ground fault and switching off the faulty line (Case 3)

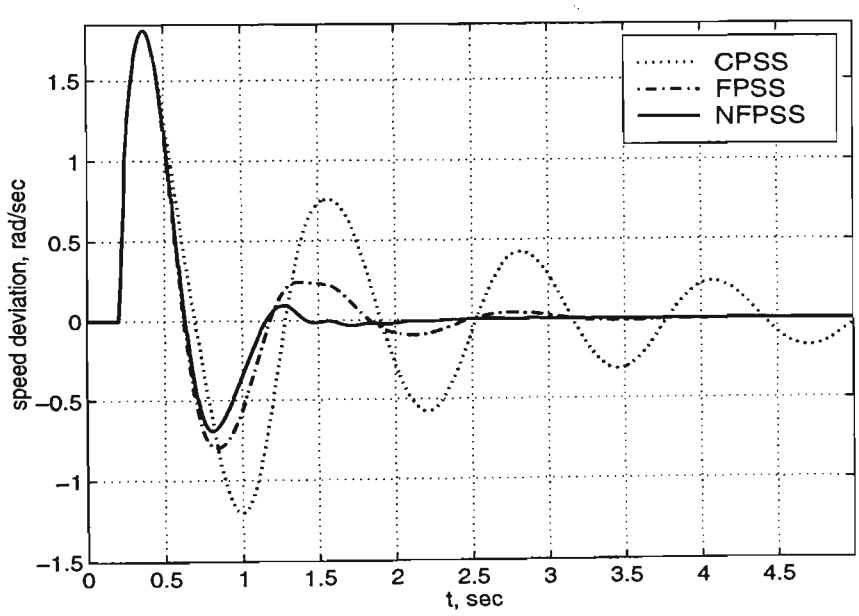


Fig. 9.41 The speed deviation response of the NFPSS compared with the CPSS and FPSS for a three-phase to ground fault and switching off the faulty line (Case 3)

9.5.1 Normal load and switching off one line

While the system was operating as in Case 1, a three-phase to ground fault

occurred at the sending end of the transmission line. The faulty line was switched off after 40 msec. The system torque angle and speed deviation are shown in Fig. 9.42

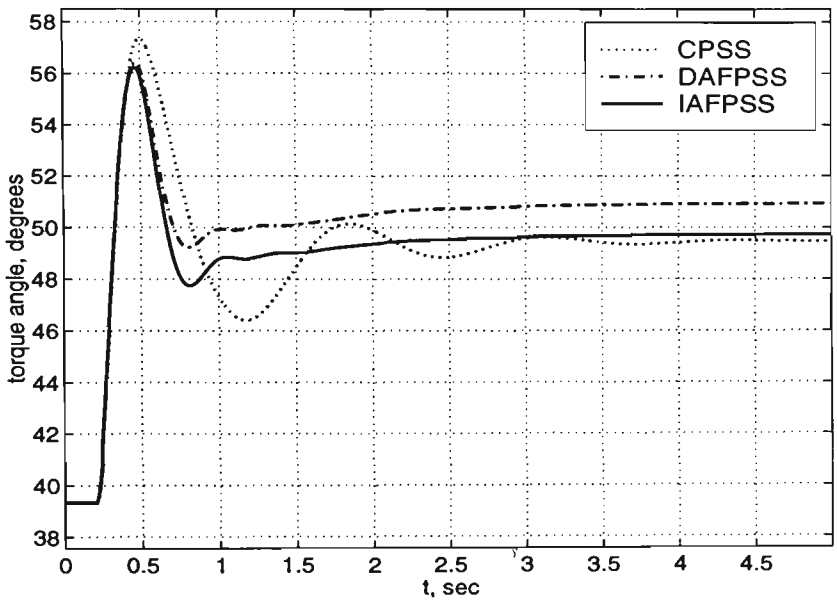


Fig. 9.42 The torque angle response of the AFPSS compared with the CPSS for a three-phase to ground fault and switching off the faulty line (Case 1)

and Fig. 9.43, respectively. J_p = 92.3 for the CPSS, J_p = 25.4 for the DAFPSS, and J_p = 27.8 for the IAFPSS. Although the speed deviation responses are very similar for the DAFPSS and the IAFPSS, the torque angle responses are different. The torque angle response of the DAFPSS has a steady state error. The reason is that although the adaptation law for the DAFPSS guarantees the convergence of the machine speed deviation to zero, it does not guarantee the stabiliser output to converge to zero. Actually it is possible that u_c in equation (8.41) have a nonzero value even for $\underline{x} = \mathbf{0}$. This fact is illustrated in Fig. 9.44 where the output of the stabilisers are compared.

This problem can be rectified by resetting the parameter vector to the value before adaptation, whenever $\|\underline{x}\| \triangleq \sqrt{x_1^2 + x_2^2}$ is smaller than a specified value such

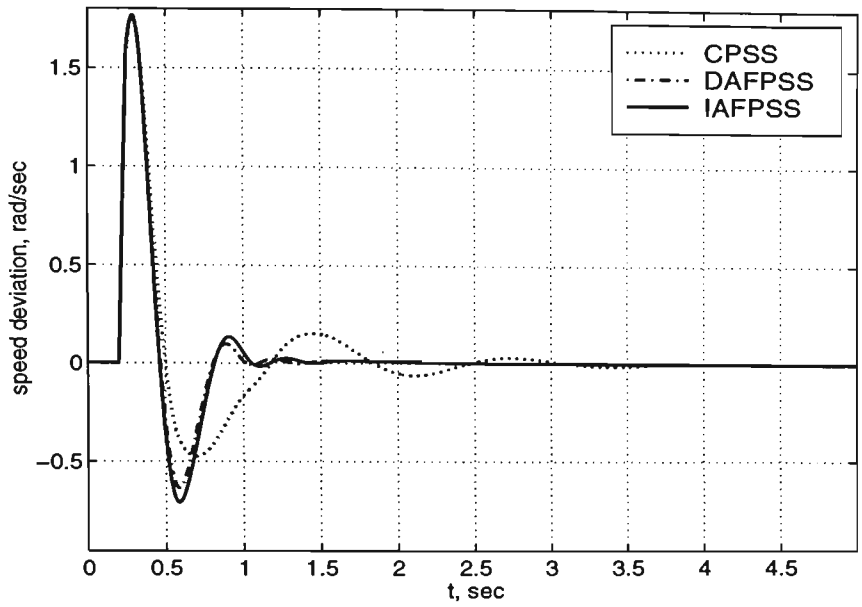


Fig. 9.43 The speed deviation response of the AFPSS compared with the CPSS for a three-phase to ground fault and switching off the faulty line (Case 1)

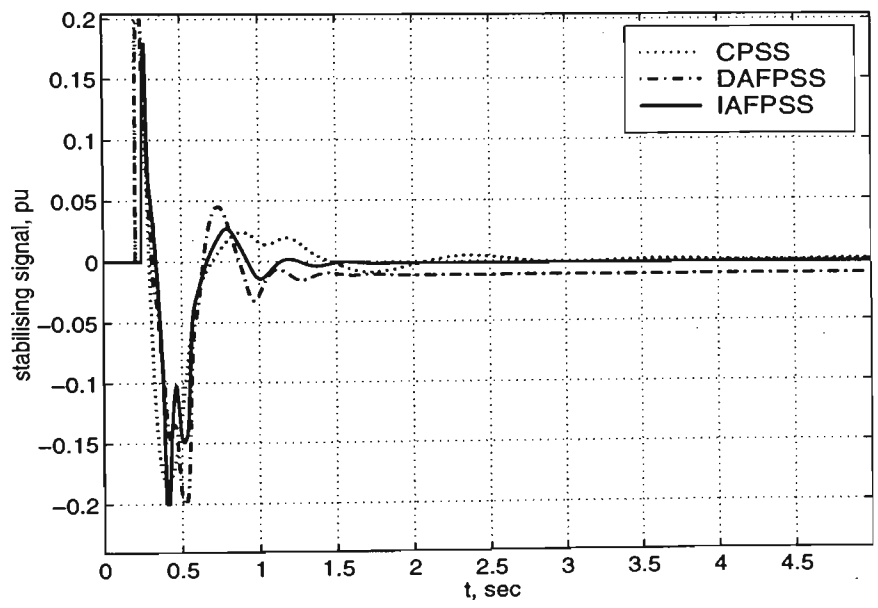


Fig. 9.44 Stabiliser output of the AFPSS compared with the CPSS for a three-phase to ground fault and switching off the faulty line (Case 1)

as 0.05. This can be done in software when the stabiliser is implemented. The torque angle response and the output of the stabilisers for the modified AFPSSs are shown in Figures 9.45 and 9.46. It can be noticed from the figures that steady state error has reduced to zero. The performance index for the modified DAFPSS is a bit higher than

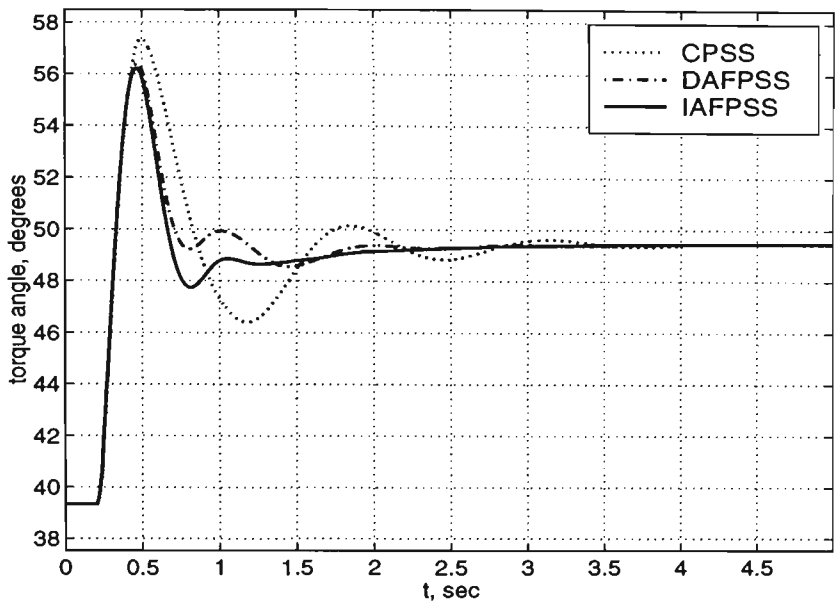


Fig. 9.45 The torque angle response of the modified AFPSSs compared with the CPSS for a three-phase to ground fault and switching off the faulty line (Case 1)

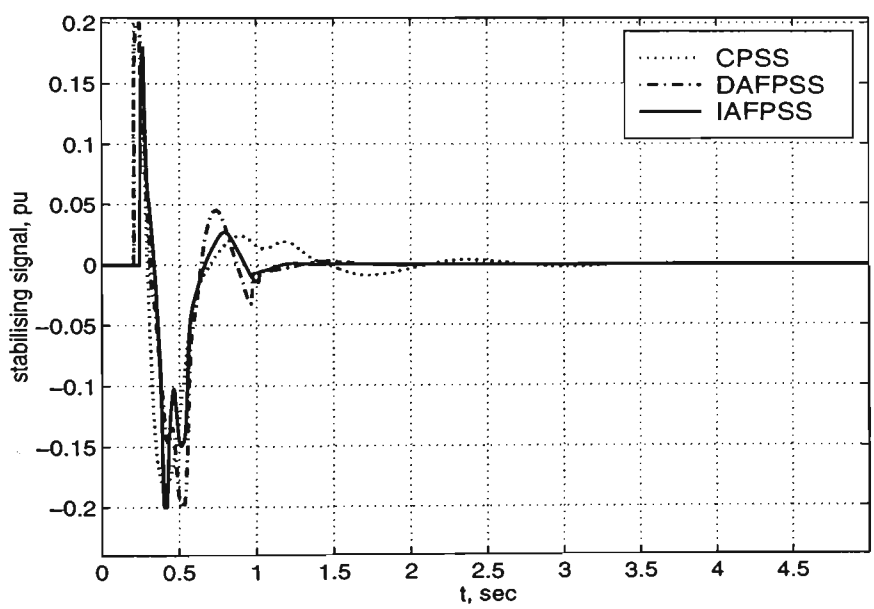


Fig. 9.46 Stabiliser output of the modified AFPSSs compared with the CPSS for a three-phase to ground fault and switching off the faulty line (Case 1)

the original DAFPSS. This performance index (J_p) is 30.6 which is a compromise for reducing the steady state error to zero. The performance index for the IAFPSS is 27.7 which has not changed.

For the rest of this chapter modified AFPSSs will be used in simulation studies.

9.5.2 Leading power factor and switching off one line

While the system was operating as in Case 2, the same fault occurred at the sending end of the transmission line. The system torque angle and speed deviation are shown in Fig. 9.47 and Fig. 9.48, respectively. J_p = 271.5 for the CPSS,

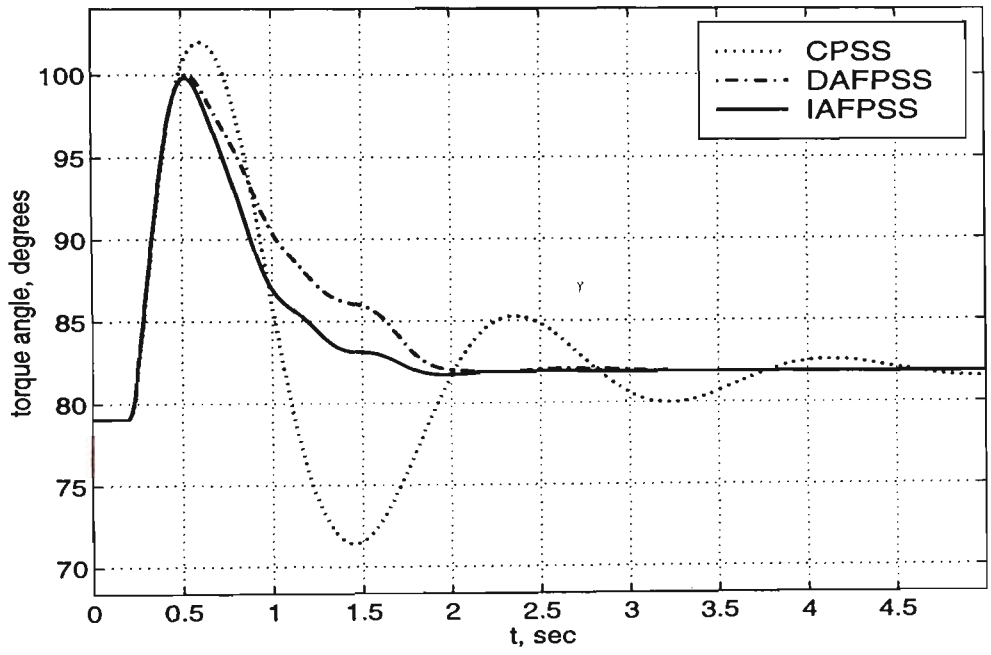


Fig. 9.47 The torque angle response of the AFPSS compared with the CPSS for a three-phase to ground fault and switching off the faulty line (Case 2)

J_p = 70.1 for the DAFPSS, and J_p = 58.3 for the IAFPSS.

9.5.3 Heavy reactive load and switching off one line

While the system was operating as in Case 3, the system was disturbed by the same fault. The system torque angle and speed deviation are shown in Fig. 9.49 and Fig. 9.50, respectively. In this Case J_p = 655 for the CPSS, J_p = 68.5 for the DAFPSS, and J_p = 110.4 for the IAFPSS.

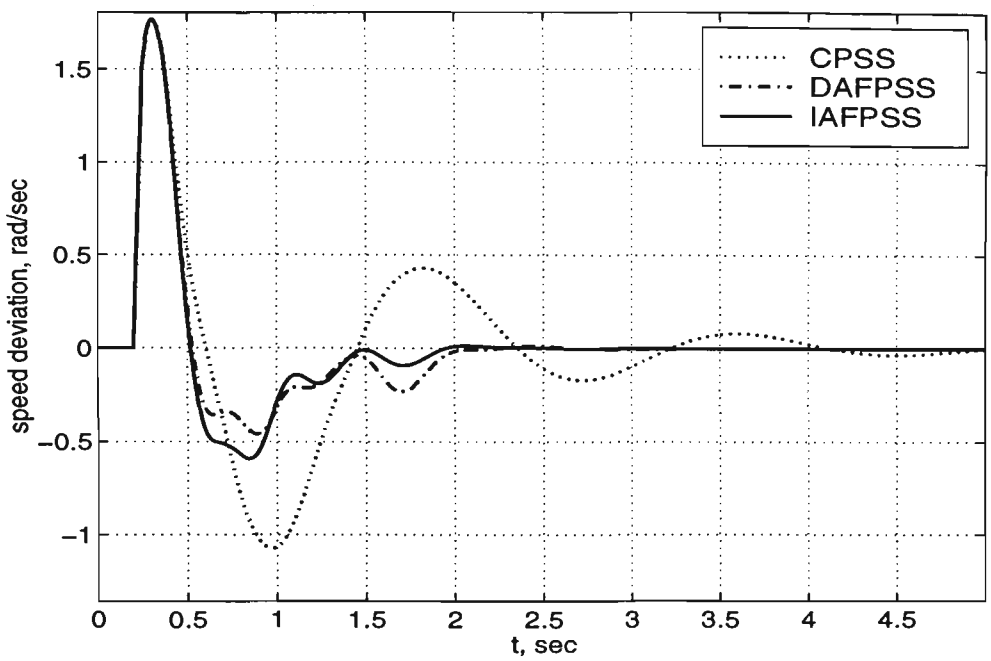


Fig. 9.48 The speed deviation response of the AFPSSs compared with the CPSS for a three-phase to ground fault and switching off the faulty line (Case 2)

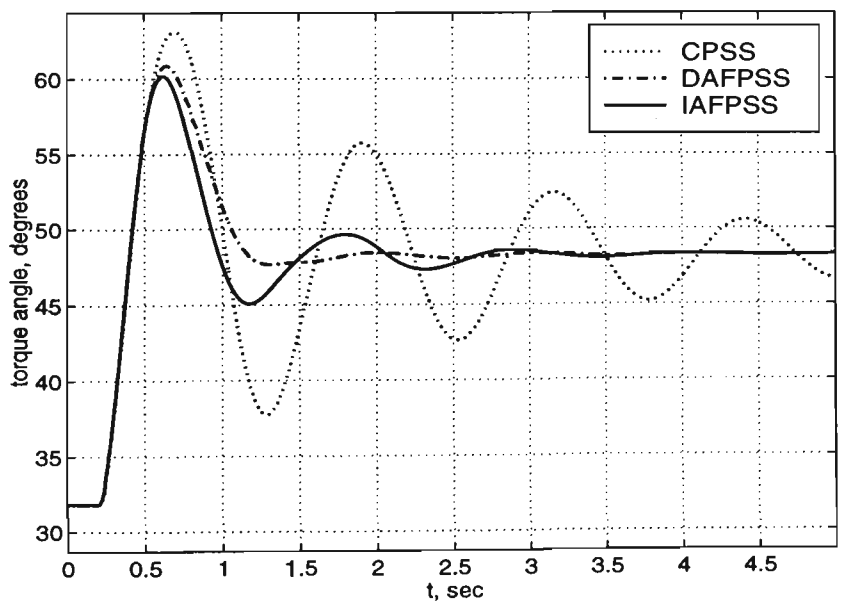


Fig. 9.49 The torque angle response of the AFPSSs compared with the CPSS for a three-phase to ground fault and switching off the faulty line (Case 3)

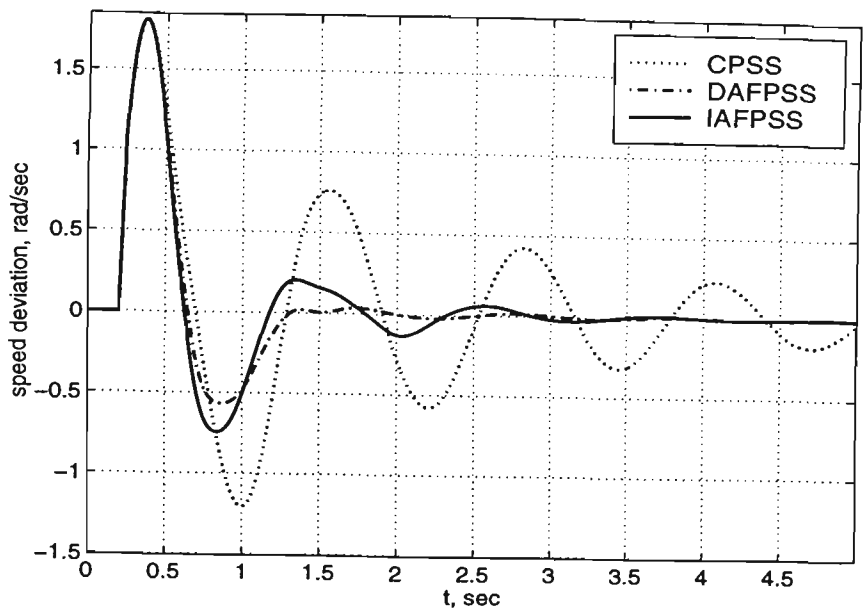


Fig. 9.50 The speed deviation response of the AFPSSs compared with the CPSS for a three-phase to ground fault and switching off the faulty line (Case 3)

9.6 Comparing the ANNPSS with the CPSS

The performance of the adaptive neural-network-based power system stabilisers (ANNPSS) is compared with the CPSS by conducting the three-phase to ground fault test as in the previous section.

9.6.1 Normal load and switching off one line

While the system was operating as in Case 1, a three-phase to ground fault occurred at the sending end of the transmission line. The faulty line was switched off after 40 msec. The system torque angle and speed deviation are shown in Fig. 9.51 and Fig. 9.52, respectively. J_p = 92.3 for the CPSS and J_p = 79.4 for the ANNPSS.

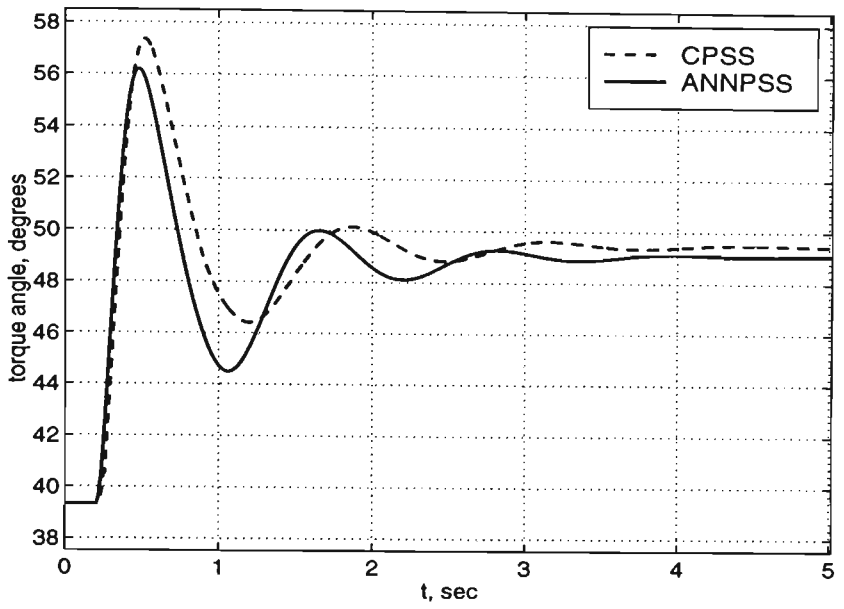


Fig. 9.51 The torque angle response of the ANNPSS compared with the CPSS for a three-phase to ground fault and switching off the faulty line (Case 1)

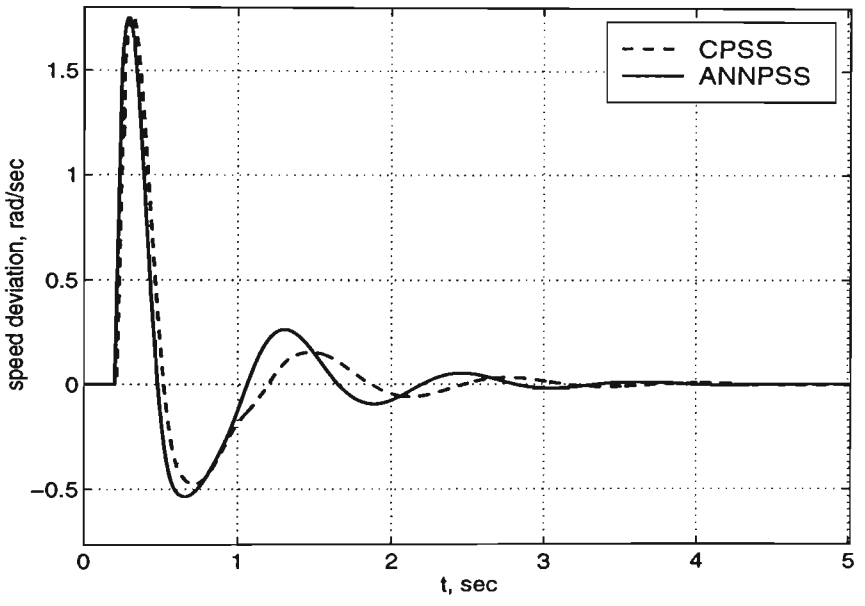


Fig. 9.52 The speed deviation response of the ANNPSS compared with the CPSS for a three-phase to ground fault and switching off the faulty line (Case 1)

9.6.2 Leading power factor and switching off one line

While the system was operating as in Case 2, the same fault occurred at the

sending end of the transmission line. The system torque angle and speed deviation are shown in Fig. 9.53 and Fig. 9.54, respectively. J_p = 271.5 for the CPSS and

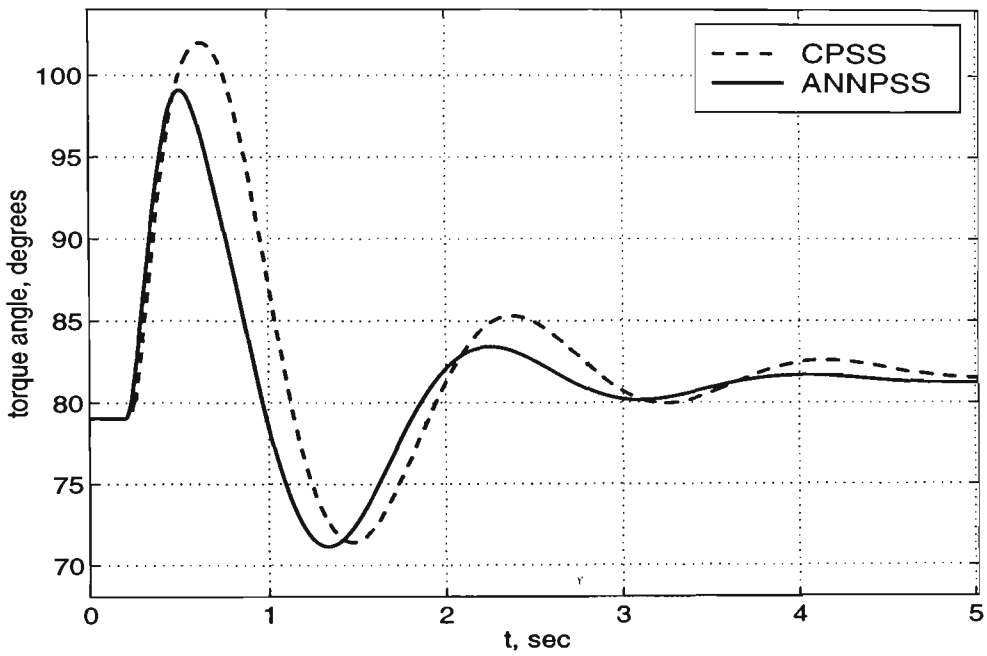


Fig. 9.53 The torque angle response of the ANNPSS compared with the CPSS for a three-phase to ground fault and switching off the faulty line (Case 2)

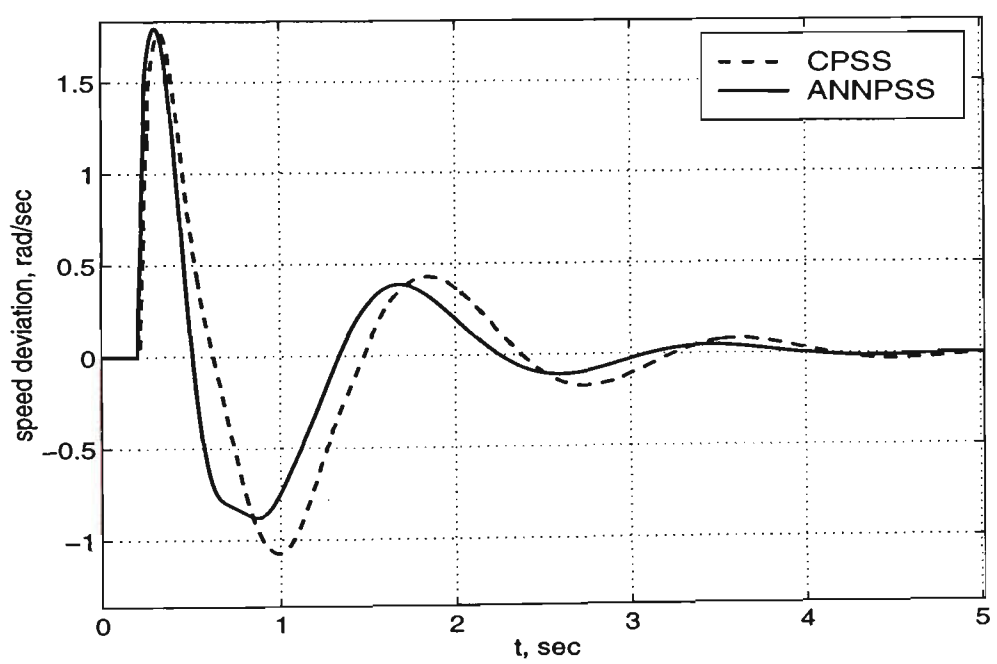


Fig. 9.54 The speed deviation response of the AFPSSs compared with the CPSS for a three-phase to ground fault and switching off the faulty line (Case 2)

J_p = 192.2 for the ANNPSS.

9.6.3 Heavy reactive load and switching off one line

While the system was operating as in Case 3, the system was disturbed by the same fault. The system torque angle and speed deviation are shown in Fig. 9.55 and

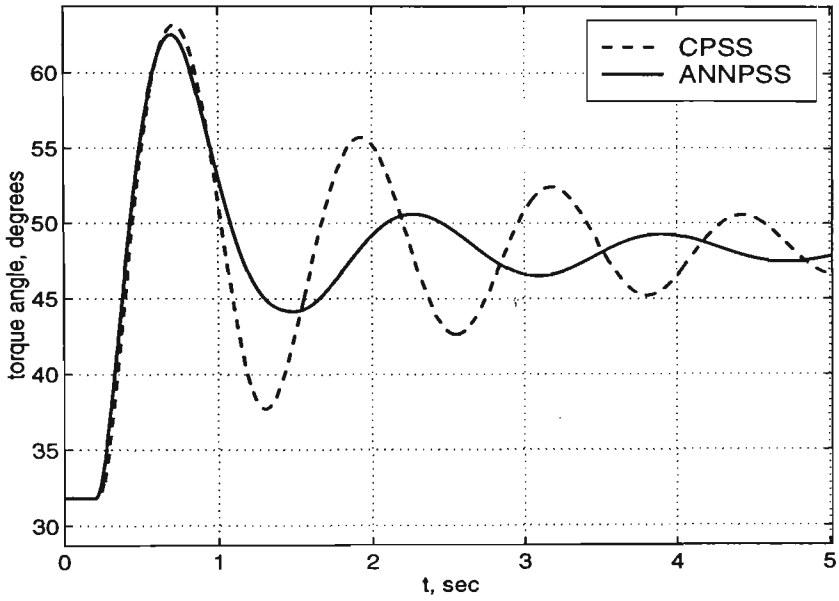


Fig. 9.55 The torque angle response of the ANNPSS compared with the CPSS for a three-phase to ground fault and switching off the faulty line (Case 3)

Fig. 9.56, respectively. J_p = 655 for the CPSS and J_p = 211 for the ANNPSS.

9.7 Comparing the performance index for the proposed PSSs

In this section the proposed PSSs are compared according to the performance index J_p obtained from the system response for five seconds after occurrence of a three-phase to ground fault. The faulty line is switched off after 40 msec. A summary of the results is given in Table 9.1. The adaptive fuzzy logic PSSs referred to in this table are the modified AFPSSs.

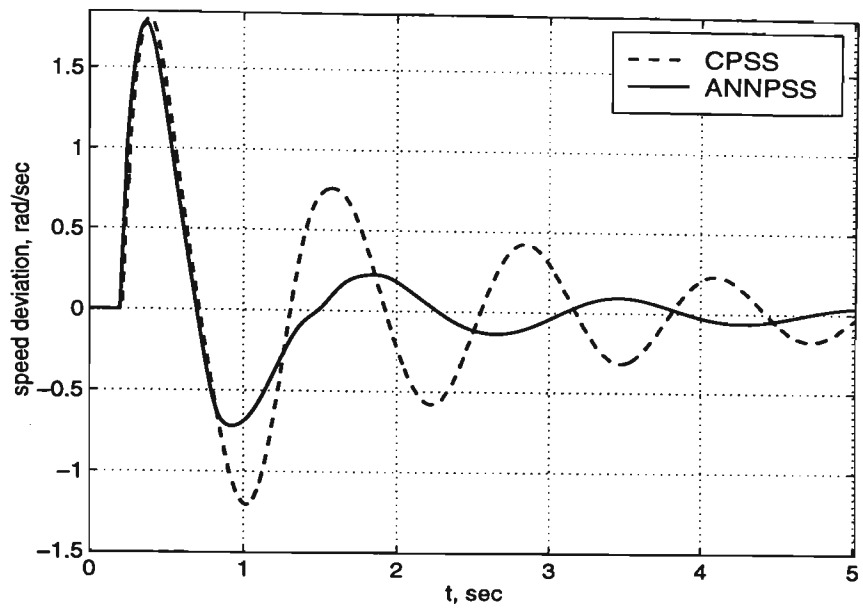


Fig. 9.56 The speed deviation response of the ANNPSS compared with the CPSS for a three-phase to ground fault and switching off the faulty line (Case 3)

Table 9.1. The performance index for the proposed PSSs

Operating Conditions	Power System Stabiliser						
	No PSS	CPSS	FPSS	NFPSS	ANNPSS	DAFPSS	IAFPSS
Case 1	1903	92.3	25.5	25.3	79.4	30.6	27.7
Case 2	5684	271.5	87.9	49.3	192.2	70.1	58.3
Case 3	2478	655	115.4	55.8	211	68.5	110.4

It is observed from the table that the NFPSS has the best performance. Performance of adaptive fuzzy PSSs are better than performance of the ANNPSS. Also performances of the adaptive PSSs are better than performance of a fixed-parameters FPSS. This is while neither of the adaptive PSSs uses accurate mathematical information regarding the power system. The AFPSSs use just some rough initial information which are far from an accurate model of the power system.

In order to show further that an AFPSS has a better performance than CPSS and FPSS in situations that the designer has not enough information about the plant, the three-phase to ground fault is applied to a different synchronous machine. The parameters of this machine is different from the one under study and are given in the Appendix C. The operating conditions are different from the previous cases, too.

The aforementioned test is done to compare the DAFPSS with the CPSS, FPSS and NFPSS for the operating conditions: $P_o = 0.85$ pu with a power factor of 0.85 lagging with the transmission line parameters $R_e = 0.03$ pu and $X_e = 0.4$ pu. The speed deviation responses are shown in Fig. 9.57. As is observed from the figure, the

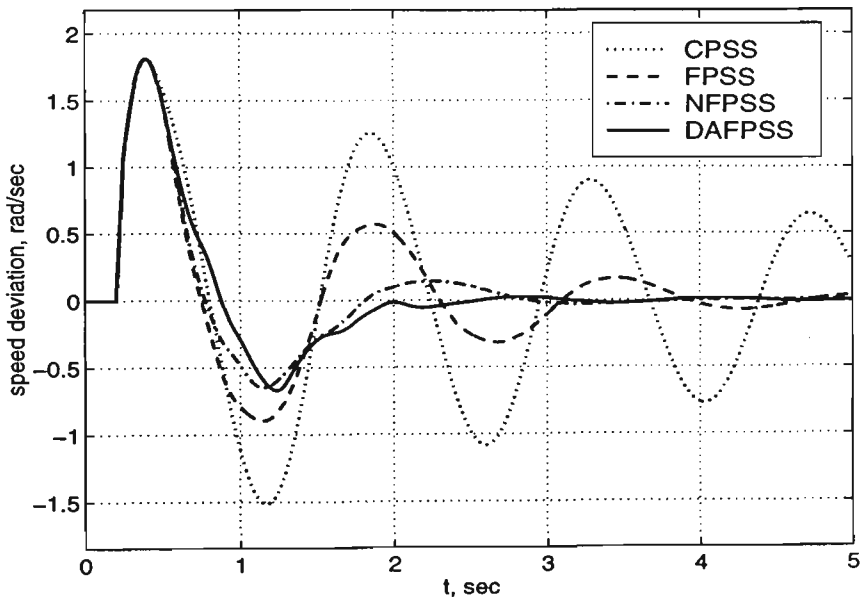


Fig. 9.57 The speed deviation response of the DAFPSS compared with the CPSS, FPSS and NFPSS for a three-phase to ground fault and switching off the faulty line

DAFPSS has the best performance. In this case $J_p = 141$ for the DAFPSS, while $J_p = 1370$ for the CPSS, $J_p = 383$ for the FPSS and $J_p = 158$ for the NFPSS.

The test is repeated for the IAFPSS. The speed deviation responses are shown in Fig. 9.58. The performance index for the IAFPSS is $J_p = 143$. The results are very close for the DAFPSS and the IAFPSS.

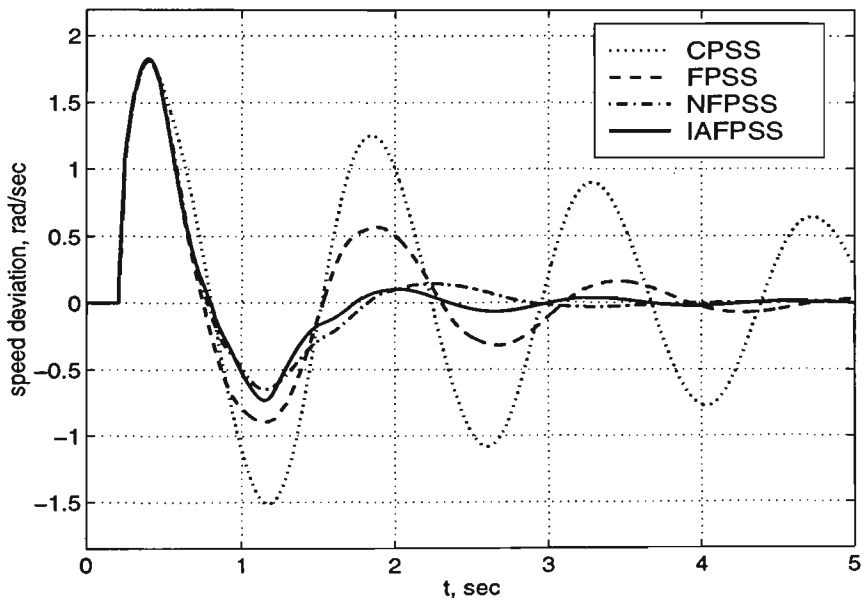


Fig. 9.58 The torque angle response of the IAFPSS compared with the CPSS, FPSS and NFPSS for a three-phase to ground fault and switching off the faulty line

9.8 Conclusions

The performances of the proposed PSSs were compared to the performance of a CPSS. It was shown that all the proposed PSSs, namely FPSS, NFPSS, ANNPSS, DAFPSS, and IAFPSS have a better performance than the CPSS. The FPSS has the simplest structure compared to the other proposed PSSs and at the same time has a good performance in a wide range of operating conditions. The NFPSS has a better performance than the FPSS. Although its structure is more complicated than the FPSS, but its complexity is reasonable. It may be considered as the best choice for sophisticated systems.

It was shown that if enough information is not available about the power system or the information is uncertain, it is possible to design AFPSSs which work satisfactorily. These PSSs are computationally more intensive than the FPSS or NFPSS.

Chapter 10

Discussions and Conclusions

10.1 Summary of Results

The primary objective and motivation for this investigation was to propose robust power system stabilisers based on fuzzy logic and neural networks. A number of PSSs were designed and investigated through nonlinear simulations.

Two types of FPSSs were designed. The first type was established based on Mamdani method for designing fuzzy controllers. The second type was designed based on the method introduced by Hiyama. These FPSSs were called fixed-parameters FPSSs, because their parameters will not change with the changes in operating conditions or system configuration. It was shown that although these fixed-parameters FPSSs have a better performance than CPSSs, their performance may degrade in some operating conditions.

In order to solve this problem of the fixed-parameters FPSSs, two types of adjustable FPSSs were proposed. In the first scheme, the FPSS is tuned on-line by using a neural network. This type of adjustable FPSS was called the neuro-fuzzy PSS (NFPSS). In the second scheme the FPSS was tuned on-line by a fuzzy logic system.

This type of adjustable FPSS was called tuned fuzzy power system stabiliser (TFPSS).

In some cases the designer may not have enough mathematical information about the system under study. It was shown that it is possible to design an adaptive PSS using neural networks just based on input-output data which is obtained on-line. This sort of PSS was called neural-network-based adaptive PSS (ANNPSS). Although it does not need a mathematical model of the system in the design procedure, its stability is not guaranteed.

This problem was the motivation to look for other adaptive PSSs which can guarantee the stability of the system. Two kinds of adaptive fuzzy PSSs were proposed. Both were designed based on the Lyapunov's synthesis method for designing stable controllers. The first type was an indirect adaptive fuzzy power system stabiliser (IAFPSS). The second type was a direct adaptive fuzzy power system stabiliser (DAFPSS). There is no need for accurate information about the system for designing such PSSs, however, having some IF-THEN rules will help in designing a better initial stabiliser. The more efficient the initial stabiliser, the more reliable the AFPSS. In other words, the more accurate IF-THEN rules available, the more reliable AFPSS.

10.2 Advantages and Disadvantages of the proposed PSSs

The FPSS is a reliable power system stabiliser. Its performance is satisfactory for a wide range of operating conditions. It is easy to implement it with microcomputers. Its implementation costs the least amongst the proposed PSSs. However, its performance depends on the operating conditions of the system - though it is more robust compared with the CPSS. For some operating conditions, the performance of the FPSS degrades to some extent.

The NFPSS and the TFPSS are optimised for different configurations and operating conditions. Therefore, they have the best performance among the proposed PSSs. However, it should be borne in mind that this optimum performance is within the expected range of operating conditions. In other words, the designer has to perform a number of tests to obtain the optimum scaling factors. These tests are done within the range that the designer expects. The broader the expected range, the more the number of required tests. If the system configuration and operating conditions change drastically, such that they are out of the expected range, the optimum performance of the NFPSS or the TFPSS is not guaranteed. Furthermore, if the designer wants to do a lot of tests in order to get a broader range for optimum response, it may be costly and cumbersome for practical systems.

From the implementation point of view, the NFPSS and the TFPSS are more complicated than the FPSS. Nonetheless, their implementation is feasible with a reasonable cost.

Amongst the three proposed adaptive PSSs, AFPSSs have a better performance compared with the ANNPSS. The advantage of APSSs is that they need the least information about the system in the design procedure. They will be adapted to the changes in the system by using the data available from input and output measurements. It was shown in simulation studies that their performances are generally better than performance of a CPSS. When the designer does not have enough information about the plant or there is a lot of uncertainty in system parameters, APSSs are a good choice.

The ANNPSS has a disadvantage that it cannot guarantee the system stability. The AFPSSs can guarantee the system stability, because they have been designed based on the Lyapunov's synthesis method. Since performances of the AFPSSs are

also better than performance of the ANNPSS, the AFPSSs are preferred compared to the ANNPSS.

The DAFPSS and the IAFPSS have similar performances. The IAFPSS is computationaly more intensive. Its implementation will be more costly. On the other hand, the DAFPSS is less complicated and its design is more straightforward. The designer can utilise the IF-THEN rules available about the PSS directly in the design of the DAFPSS. Therefore, the DAFPSS may be preferred compared to the IAFPSS.

10.3 Fulfilment of the Objectives Outlined in the Introduction

- It was shown that if enough mathematical information is not available about the plant, fuzzy logic or artificial neural networks can be used to design effective PSSs.

- If some linguistic IF-THEN rules are available from the experts they can be used to design reliable FPSSs.

- An optimum PSS can be designed by combining fuzzy logic and artificial neural networks to operate in a wide range of operating conditions, as it was explained in the design of the NFPSS.

- The FPSS can be tuned on-line using artificial neural networks (the NFPSS) or another fuzzy logic system (the TFPSS).

- Hierarchical structure of artificial neural networks can be used to design adaptive PSSs.

- Stable fuzzy logic adaptive PSSs can be designed using Lyapunov's synthesis method (as explained in the design procedure of AFPSSs).

- Performances of the proposed PSSs based on fuzzy logic or/and neural networks were compared using a performance index. Advantages and disadvantages of the

proposed schemes were discussed.

10.4 Future Research

The research can be continued in two ways:

Firstly, the simulation studies can be extended for multi-machine power systems. The simulation studies performed in this thesis were conducted in a single machine infinite bus environment. The reason was the fact that the proposed PSSs were aimed to be used in an industrial cogenerator system. From the cogenerator side, the rest of the power system can be considered as an infinite bus. Furthermore, the objective of this research was to concentrate on the design of various PSSs using fuzzy logic and neural networks rather than power system dynamic analysis. Nevertheless, the ability of the proposed PSSs to damp multi-mode oscillations in the multi-machine environment needs to be verified. Multi-mode oscillations consist of three modes of oscillations: a) Inter-Machine mode which describes frequencies related to closely coupled generators swinging relative to each other. b) Local mode which refers to oscillations occurring in plant transients caused by generator rotors oscillating relative to the combined equivalent inertia of the system. c) Inter-Area modes which are the frequencies caused by the coherent groups of generators in one area swinging relative to a number of other coherent groups in other areas. It is recommended that only three of the proposed PSSs be considered for these simulation studies, i.e., the FPSS, the NFPSS and the DAFPSS. Other proposed schemes were shown to be either similar or worse than these three PSSs.

Secondly, the research can be continued by implementing the NFPSS and DAFPSS. They should be tested practically in different stages. At first they can be tested using a laboratory-scale generator. Then they can be tested in the field by applying them to an industrial cogenerator or any practical generator.

References

[1] P. Kundur, "Power system stability and control", McGraw-Hill Inc., U.S.A., 1994.

[2] P. M. Anderson and A. A. Fouad, "Power system control and stability", Iowa State University Press, Iowa, U.S.A., 1977.

[3] C. Concordia and P. G. Brown, "Effects of trends in large steam turbine driven generator parameters on power system stability", IEEE Trans. on Power Apparatus and Systems, vol. PAS-90, Sept. 1971, pp. 2211-2218.

[4] C. P. Steinmetz, "Power control and stability of electric generating stations", AIEE Transactions, July-December 1920, p. 1215.

[5] R. D. Evans and R. C. Bergvall, "Experimental analysis of stability and power limitations", AIEE Transactions, 1924, pp. 39-58.

[6] R. D. Evans and C. F. Wagner, "Further studies of transmission system stability", AIEE Transactions, 1926, pp. 51-80.

[7] R. Wilkins, "Practical aspects of system stability", AIEE Transactions, 1926, pp. 41-50.

[8] AIEE Subcommittee on Interconnections and Stability Factors, "First report of power system stability", AIEE Transactions, February 1937, pp. 261-282.

[9] S. B. Crag, "Long distance power transmission", AIEE Trans. (pt. 2), 1950, pp. 834-844.

[10] F. P. DeMello and C. Concordia, "Concepts of synchronous machine stability as affected by excitation control", IEEE Trans. on Power Apparatus and Systems, vol. PAS-88, April 1969, pp. 316-329.

[11] F. P. DeMello and T. F. Laskowski, "Concepts of power system dynamic stability", IEEE Trans. on Power Apparatus and Systems, vol. PAS-94, May/June 1975, pp. 827-833.

[12] M. K. El-Sherbiny and A. A. Fouad, "Digital analysis of excitation control for interconnected power systems", IEEE Trans. on Power Apparatus and Systems, vol. PAS-88, 1969, pp. 316-329.

[13] P. G. Brown, F. P. DeMello and E. H. Lenfest, "Effects of excitation, turbine energy control, and transmission on transient stability", IEEE Trans. on Power Apparatus and Systems, vol. PAS-89, July/August 1970, pp. 1247-1252.

[14] O. L. Elgred, "Electric Energy Systems Theory - An introduction", McGraw-Hill Inc., 1985.

[15] S. N. Iyer and B. J. Cory, "Optimal control of turbo-generator including an exciter and governor", IEEE Trans. on Power Apparatus and Systems, vol. PAS-90, Sept./Oct. 1971, pp. 2142-2149.

[16] W. A. Mittelstadt, "Four methods of power system damping", IEEE Trans. on Power Apparatus and Systems, vol. PAS-87, May 1968, pp. 1323-1329.

[17] O. J. Smith, "Power system transient control by capacitor switching", IEEE Trans. on Power Apparatus and Systems, vol. PAS-88, Jan. 1969, pp. 28-35.

[18] E. V. Larsen and D. A. Swann, "Applying power system stabilizers, Part I: General concepts", IEEE Trans. on Power Apparatus and Systems, vol. PAS-100, no. 6, June 1981, pp. 3017-3024.

[19] E. V. Larsen and D. A. Swann, "Applying power system stabilizers, Part II: Performance objectives and tuning concepts", IEEE Trans. on Power Apparatus and Systems, vol. PAS-100, no. 6, June 1981, pp. 3025-3033.

[20] E. V. Larsen and D. A. Swann, "Applying power system stabilizers, Part III: Practical considerations", IEEE Trans. on Power Apparatus and Systems, vol. PAS-100, no. 6, June 1981, pp. 3034-3046.

[21] M. K. El-Sherbiny and D. M. Mehta, "Dynamic system stability - part I: Investigation of different loading and excitation systems", IEEE Trans. on Power Apparatus and Systems, vol. PAS-92, Sept. /Oct. 1973, pp. 1538-1546.

[22] C. Concordia, "Steady state stability of synchronous machines as affected by voltage regulator characteristics", AIEE Trans. on Power Apparatus and Systems, vol. PAS-63, 1944, pp. 215-220.

[23] P. L. Dandeno, A. N. Karas, K. R. McClymont and W. Watson, "Effect of high-speed rectifier excitation systems on generator stability limits", IEEE Trans. on Power Apparatus and Systems, vol. PAS-87, Jan. 1968, pp. 190-201.

[24] O. W. Hanson, C. J. Goodwin and P. L. Dandeno, "Influence of excitation and speed control parameters in stabilizing inter-system oscillations", IEEE Trans. on Power Apparatus and Systems, vol. PAS-87, May 1968, pp. 1306-1313.

[25] F. R. Schlief, H. D. Hunkins, G. E. Martin and E. E. Hattan, "Excitation control to improve power line stability", IEEE Trans. on Power Apparatus and Systems vol. PAS-87, no. 6, June 1968, pp. 1426-1434.

[26] F. W. Keay and W. H. South, "Design of a power system stabilizer sensing frequency deviation", IEEE Trans. on Power Apparatus and Systems, vol. PAS-90, no. 2, March/April 1971, pp. 707-713.

[27] W. Watson and M. E. Coultes, "Static exciter stabilizing signals on large generators - mechanical problems", IEEE Trans. on Power Apparatus and Systems, vol. PAS-92, Jan./Feb. 1973, pp. 204-212.

[28] F. P. DeMello, L. N. Hannet, D. W. Parkinson and J. S. Czuba, "A power system stabilizer design using digital control", IEEE Trans. on Power Apparatus and Systems, vol. PAS-101, 1982, pp. 2860-2868.

[29] F. P. DeMello, L. N. Hannet and J. M. Undrill, "Practical approaches to supplementary stabilizing from accelerating power", IEEE Trans. on Power Apparatus and Systems, vol. PAS-97, 1978, pp. 1515-1522.

[30] F. P. DeMello, P. J. Nolan, T. F. Laskowski and J. M. Undrill, "Coordinated application of stabilisers in multi-machine power systems", IEEE Trans. on Power Apparatus and Systems, vol. PAS-99, 1980, pp. 892-901.

[31] F. R. Schleif, C. G. Martin and R. R. Angell, "Damping of system oscillations with hydro-generating unit", IEEE Trans. on Power Apparatus and Systems vol. PAS-86, no. 4, April 1967, pp. 438-442.

[32] W. Watson and G. Manchur, "Experience with supplementary damping signal for generator static excitation systems", IEEE Trans. on Power Apparatus and Systems, vol. PAS-92, Jan./Feb. 1973, pp. 199-203.

[33] J. P. Bayne, D. C. Lee and W. Watson, "A power system stabilizer for thermal units based on derivation of accelerating power", IEEE Trans. on Power Apparatus and Systems, vol. PAS-96, no. 6, Dec. 1977, pp. 1777-1783.

[34] D. C. Lee, R. E. Beaulieu and J. R. R. Service, "A power system stabilizer using speed and electrical power inputs, design and field experience", IEEE Trans. on Power Apparatus and Systems, vol. PAS-100, no. 9, Sept. 1981, pp. 4151-4157.

[35] IEEE Committee Report, "Computer representation of excitation system", IEEE Trans. on Power Apparatus and Systems, vol. PAS-87, no. 6, June 1968, pp. 1460-1464.

[36] N. Hosseinzadeh, A. Kalam and W. S. Lee, "A comparison study between a conventional and a fuzzy logic based power system stabiliser for a synchronous generator", 10th International Power System Conference, 6-8 Nov. 1995, Tehran, Iran, pp. 183-191.

[37] R. J. Fieming, M. A. Mohan and K. Parvatisam, "Selection of parameters of stabilisers in multi-machine power system", IEEE Trans. on Power Apparatus and Systems, vol. PAS-100, no. 5, May 1981, pp. 2329-2333.

[38] Y. N. Yu and H. A. M. Moussa, "Optimal stabilization of a multi-machine system", IEEE Trans. on Power Apparatus and Systems, vol. PAS-91, no. 3, May/June 1972, pp. 1174-1182.

[39] W. C. Chan and Y. Y. Hsu, "An optimal variable structure stabilizer for power system stabilization", IEEE Trans. on Power Apparatus and Systems, vol. PAS-102, no. 6, June 1983, pp. 1738-1746.

[40] R. J. Koessler, "Techniques for tuning excitation system parameters", IEEE Trans. on Energy Conversion, vol. 3, no. 4, December 1988, pp. 785-791.

[41] K. Bhattacharya, J. Nanda and M. L. Kothari, "Optimization and performance analysis of conventional power system stabilizers", International Journal of Electrical Power and Energy Systems, vol. 19, no. 7, Oct. 1997, pp. 449-458.

[42] Y. Y. Hsu, C. S. Liu, T. S. Luor, C. L. Chang, A. S. Liu, Y. T. Chen and C. T. Huang, "Experience with the identification and tuning of excitation system parameters at the second nuclear power plant of Taiwan power company", IEEE Trans. on Power Systems, vol. 11, no. 2, May 1996, pp. 747-753.

[43] A. Doi and S. Abe, "Coordinated synthesis of power system stabilizers in multi-machine power systems", IEEE Trans. on Power Apparatus and Systems, vol. PAS-103, no. 6, June 1984, pp. 1473-1479.

[44] K. Othsuka, S. Yokokama, H. Tanaka and H. Doi, "A multivariable optimal control system for a generator", IEEE Trans. on Energy Conversion, vol. EC-1, June 1986, pp. 88-98.

[45] M. L. Kothari, J. Nanda and K. Bhattacharya, "Co-ordinated tuning of power system stabilizers for a multi-machine system by sequential application of ISE technique", APSCOM'93- Second Int. Conference on Advances in Power System Control, Operation and Management, Hong Kong, Dec. 1993, vol. 1, pp. 299-304.

[46] T. C. Yang, "Applying structured singular value to multi-machine power system

stabilizer design", Electric Power Systems Research, vol. 43, no. 2, Nov. 1997, pp. 113-123.

[47] K. E. Bollinger and S. Z. Ao, "PSS performance as affected by its output limiter", IEEE Trans. on Energy Conversion, vol. 11, no. 1, March 1996, pp. 118-124.

[48] M. Saidy and F. M. Hughes, "Performance improvement of a conventional power system stabilizer", International Journal of Electrical Power and Energy Systems, vol. 17, no. 5, Oct. 1995, pp. 313-323.

[49] C. Wen and M. J. Gibbard, "Conventional power system stabilizer with auxiliary self tuning fixed-parameter controller", International Journal of Electrical Power and Energy Systems, vol. 17, no. 1, Feb. 1995, pp. 39-49.

[50] M. L. Kothari, J. Nanda and K. Bhattacharya, "Discrete mode power system stabilisers", IEE Proceedings-C, vol. 140, no. 6, November 1993, pp. 523-531.

[51] B. Habibullah and Y. N. Yu, "Physically realizable wide power range optimal controllers for power systems", IEEE Trans. on Power Apparatus and Systems, vol. PAS-93, Sept./Oct. 1974, pp. 1498-1506.

[52] M. M. Elmetwally, N. D. Rao and O. P. Malik, "Experimental results on the implementation of an optimal control for synchronous machines", IEEE Trans. on Power Apparatus and Systems, vol. PAS-94, July/Aug. 1975, pp. 1192-1200.

[53] M. R. Khaldi, A. K. Sarkar, K. Y. Lee and Y. M. Park, "The modal performance measure for parameter optimization of power system stabilizers", IEEE Trans. on Energy Conversion, vol. 8, no. 4, Dec. 1993, pp. 660-666.

[54] A. J. A. S. Costa, F. D. Freitas and H. E. Pena, "Power system stabilizer design via structurally constrained optimal control", Electric Power Systems Research, vol. 33, no. 1, April 1995, pp. 33-40.

[55] Y. L. Abdelmagid and M. M. Dawoud, "Tuning of power system stabilizers using genetic algorithms", Electric Power Systems Research, vol. 39, no. 2, Nov. 1996, pp. 137-143.

[56] K. J. Astrom and B. Wittenmark, "Adaptive control", Addison-Wesley Publishing Company, Reading, Mass., 1989.

[57] D. Xia and G. T. Heydt, "Self-Tuning controller for generator excitation control", IEEE Trans. on Power Apparatus and Systems, vol. PAS-102, no. 6, June 1983, pp. 1877-1885.

[58] S. J. Cheng, Y. S. Chow, O. P. Malik, and G. S Hope, "An adaptive synchronous machine stabilizer", IEEE Trans. on Power Systems, vol. 1, no. 3, 1986, pp 101-109.

[59] C. J. Wu and Y. Y. Hsu, "Design of self-tuning PID power system stabilizer for multimachine power systems", IEEE Trans. on Power Systems, vol. 3, no. 3, August 1988, pp. 1059-1064.

[60] A. Chandra, O. P. Malik and G. S. Hope, "A self-tuning controller for the control of multi-machine power systems", IEEE Trans. on Power Systems, vol. 3, no. 3, August 1988, pp. 1065-1071.

[61] J. W. Finch, K. J. Zachariah and M. Farsi, "Self-tuning control applied to turbogenerator AVRs", IEE Proc. on Generation, Transmission and Distribution, vol. 143, no. 5, Sept. 1996, pp 492-499.

[62] G. P. Chen and O. P. Malik, "Tracking constrained adaptive power system stabiliser", IEE Proc. -Gener. Transm. Distrib., vol. 142, no. 2, March 1995, pp. 149-156.

[63] O. P. Malik, "Adaptive control of synchronous machine excitation", in Microprocessor-Based Control Systems, N. K. Sinha (ed.), D. Reidel Publishing Company, 1986, pp. 61-79.

[64] Y. Y. Hsu and C. J. Wu, "Adaptive control of a synchronous machine using the auto-searching method", IEEE Trans. on Power Systems, vol. 3, no. 4, Nov. 1988, pp. 1434-1440.

[65] N. C. Pahalawatha, G. S. Hope, O. P. Malik and K. Wong, "Real time implementation of a MIMO adaptive power system stabiliser", IEE Proceedings, vol. 137, Pt. C, no. 3, May 1990, pp. 186-194.

[66] O. P. Malik, G. P. Chen, G. S. Hope, Y. H Qin and G. Y. Xu, "Adaptive self-optimizing pole shifting control algorithm", IEEE Trans. on Energy Conversion, vol. 8, no. 4, 1993, pp. 639-645.

[67] M. L. Kothari, K. Bhattacharya and J. Nanda, "Adaptive power system stabiliser based on pole-shifting technique", IEE Proc. -Gener. Transm. Distrib., vol. 143, Pt. C, no. 1, Jan. 1996, pp. 96-98.

[68] Y. M. Park and W. Kim, "Discrete-time adaptive sliding mode power system stabilizer with only input/output measurements", International Journal of Electrical Power and Energy Systems, vol. 18, no. 8, Nov. 1996, pp. 509-517.

[69] F. P. He and M. J. Gibbard, "Design of an adaptive bilinear power system stabilizer", Automatica, vol. 33, no. 4, Apr. 1997, pp. 663-668.

[70] R. Asgharian, "A robust H_∞ power system stabilizer with no adverse effect on shaft torsional modes", IEEE Trans. on Energy Conversion, vol. 9, no. 3, 1994, pp. 475-481.

[71] Q. H. Zhao and J. Jiang, "Robust controller design for generator excitation systems", IEEE Trans. on Energy Conversion, vol. 10, no. 2, June 1995, pp. 201-209.

[72] G. N. Taranto, J. H. Chow and H. A. Othman, "Robust redesign of power system damping controllers", IEEE Trans. on Control Systems Technology, vol. 3, no. 3, Sept. 1995, pp. 290-298.

[73] R. Asgharian and S. A. Tavakoli, "A systematic approach to performance weights selection in design of robust H_∞ PSS using genetic algorithms", IEEE Trans. on Energy Conversion, vol. 11, no. 1, March 1996, pp. 111-117.

[74] R. Huwer, P. Kocher, D. Nelles and M. Wache, "Design of a linear robust PSS for optimal damping of multimode oscillations", European Trans. on Electrical Power, vol. 6, no. 1, Jan-Feb 1996, pp. 67-70.

[75] S. Chen and O. P. Malik, "H_∞ optimisation-based power system stabiliser design", IEE Proc. -Gener. Transm. Distrib., vol. 142, no. 2, March 1995, pp. 179-184.

[76] S. Chen and O. P. Malik, "Power system stabilizer design using μ synthesis", IEEE Trans. on Energy Conversion, vol. 10, no. 1, March 1995, pp. 175-181.

[77] S. Chen, O. P. Malik and T. W. Chen, "A robust power system stabilizer design", Optimal Control Applications and Methods, vol. 18, no. 3, May-June 1997, pp. 179-193.

[78] K. A. Folly, N. Yorino and H. Sasaki, "Design of H-infinity-PSS using numerator-denominator uncertainty representation", IEEE Trans. on Energy Conversion, vol. 12, no. 1, March 1997, pp. 45-50.

[79] G. Guo, Y. Wang, K. Y. Lim and L. Gao, "Robust nonlinear controller for power system transient stability enhancement with voltage regulation", IEE Proc. on Generation, Transmission and Distribution, vol. 143, no. 5, Sept. 1996, pp 407-412.

[80] V. G. D. C. Samarasinghe and N. C. Pahalawatha, "Stabilization of a

multi-machine power system using nonlinear robust variable structure control", Electric Power Systems Research, vol. 43, no. 1, Oct. 1997, pp. 11-17.

[81] J. C. Doyle, "Analysis of feedback systems with structured uncertainty", Proc. IEE-part D, vol. 129, Jan. 1982, pp. 242-250.

[82] J. C. Doyle, K. Glover, P. P. Khargonekar and B. A. Francis, "State-Space solution to standard H_2 and H_∞ control problems", IEEE Trans. on Automatic Control, vol. AC-34, no. 8, Aug. 1989, pp. 831-847.

[83] P. Dorato and R. K. Yedavalli, "Recent advances in robust control", IEEE Press, New York, 1990.

[84] T. Lahdhiri and A. T. Alouani, "Nonlinear stabilizing controller for a single machine infinite-bus power system", Proceedings of the 4th IEEE Conference on Control Applications, Albany, NY, USA, Sept. 1995, pp. 1014-1019.

[85] C. Sun, Z. Zhao, Y. Sun and Q. Lu, "Design of nonlinear robust excitation control for multimachine power systems", IEE Proc. on Generation, Transmission and Distribution, vol. 143, no. 3, May 1996, pp 253-257.

[86] A. M. Sharaf and T. T. Lie, "An elastic nonlinear power system stabilizer", Electric Machines and Power Systems, vol. 24, no. 8, Dec. 1996, pp. 857-867.

[87] M. Nambu and Y. Ohsawa, "Development of an advanced power system stabilizer using a strict linearization approach", IEEE Trans. on Power Systems, vol. 11, no. 2, May 1996, pp. 813-818.

[88] Y. Y. Hsu and C. R. Chen, "Tuning of power system stabilizer using an artificial neural network", IEEE Trans. on Energy Conversion, vol. 6, Dec. 1991, pp. 612-619.

[89] Q. H. Wu, B. W. Hogg and G. W. Irwin, "A neural network regulator for turbo-generators", IEEE Trans. on Neural Networks, vol. 3, Jan. 1992, pp. 95-100.

[90] Y. Zhang, G. P. Chen, O. P. Malik and G. S. Hope, "An artificial neural network based adaptive power system stabilizer", IEEE Trans. on Energy Conversion, vol. 8, March 1993, pp. 71-77.

[91] J. M. Mendel, "Fuzzy logic systems for engineering: a tutorial", Proceedings of the IEEE, vol. 83, no. 3, March 1995, pp. 345-377.

[92] L. X. Wang, "Fuzzy systems are universal approximators", Proc. IEEE

International Conference on Fuzzy Systems, San Diego, CA, 1992, pp. 1163-1170.

[93] L. A. Zadeh, "Fuzzy sets", Information Control, vol. 8, 1965, pp. 338-353.

[94] C. C. Lee, "Fuzzy logic in control systems: fuzzy logic controller", IEEE Trans. on Systems, Man and Cybernetics, vol. 20, no. 2, March/April 1990, pp. 404-435.

[95] L. X. Wang, "Adaptive fuzzy systems and control: Design and stability analysis", Prentice-Hall, Englewood Cliffs, NJ, 1994.

[96] M. A. M. Hassan, O. P. Malik and G. S. Hope, "A fuzzy logic based stabilizer for a synchronous machine", IEEE Trans. on Energy Conversion, vol. 6, no. 3, 1991, pp. 407-413.

[97] T. Hiyama, "Application of rule-based stabiliser controller to electric power system", IEE Proc. -Gener. Transm. Distrib., vol. 136, no. 3, 1989, pp. 175-181.

[98] J. Shi, L. H. Herron and A. Kalam, "Design and implementation of a fuzzy logic based power system stabilizer", AMSE Periodicals: Advances in Modelling and Analysis, C. AMSE Press, vol. 42, no. 4, 1994, pp. 39-52.

[99] J. Shi, L. H. Herron and A. Kalam, "Optimization of fuzzy controllers as real-time power system stabilizers", Special issue on "Control Applications of Fuzzy Logic", International Journal for Engineering Applications of Artificial Intelligence, Pergamon Press, vol. 7, no. 5, October 1994, pp. 545-558.

[100] K. A. El-Metwally and O. P. Malik, "Fuzzy logic power system stabiliser", IEE Proc. on Generation, Transmission and Distribution, vol. 142, no. 3, May 1995, pp 277-281.

[101] K. A. El-Metwally, G.C. Hancock and O.P. Malik, "Implementation of a fuzzy logic PSS using a micro-controller and experimental test results", IEEE Trans. on Energy Conversion, vol. 11, no. 1, March 1996, pp. 91-96.

[102] T. Hiyama, "Real time control of micro-machine system using micro-computer based fuzzy logic power system stabilizer", IEEE Trans. on Energy Conversion, vol. 9, no. 4, Dec. 1994, pp 724-731.

[103] Y. Y. Hsu and C. H. Cheng, "Design of fuzzy power system stabilisers for multimachine power systems", IEE Proc. -Gener. Transm. Distrib., vol. 137, Pt. C, no. 3, May 1990, pp. 233-238.

[104] M. Parniani and H. Lesani, "Application of power system stabilizer at Bandar-Abbas power station", IEEE Trans. on Power Systems, vol. 9, no. 3, Aug. 1994, pp. 1366-1370.

[105] T. H. Ortmeyer and T. Hiyama, "Frequency response characteristics of the fuzzy polar power system stabilizer", IEEE Trans. on Energy Conversion, vol. 10, no. 2, June 1995, pp. 333-338.

[106] T. Hiyama, K. Miyazaki and H. Satoh, "A fuzzy logic excitation system for stability enhancement of power systems with multi-mode oscillations", IEEE Trans. on Energy Conversion, vol. 11, no. 2, June 1996, pp. 449-454.

[107] P. Hoang and K. Tomsovic, "Design and analysis of an adaptive fuzzy power system stabilizer", IEEE Trans. on Energy Conversion, vol. 11, no. 2, June 1996, pp. 455-461.

[108] M. K. Elsherbiny, G. Elsaady, E. A. Ibrahim and A. M. Sharaf, "Efficient incremental fuzzy logic controller for power system stabilization", Electric Machines and Power Systems, vol. 25, no. 4, May 1997, pp. 429-441.

[109] H. A. Toliyat, J. Sadeh and R. Ghazi, "Design of augmented fuzzy logic power system stabilizers to enhance power systems stability", IEEE Trans. on Energy Conversion, vol. 11, no. 1, March 1996, pp. 97-103.

[110] O. Ghanayem and L. Reznik, "A hybrid AVR-PSS controller based on fuzzy logic technique", CONTROL'95 Conference, Melbourne, Australia, 1995, vol. 2, pp. 347-351.

[111] M. M. Pedram and H. Seifi, "Extended algorithm of a fuzzy-set based power system stabilizer with genetic algorithm tuning", European Transactions on Electrical Power, vol. 7, no. 3, May-June 1997, pp. 205-210.

[112] P. J. Antsaklis, "Neural networks in control systems", IEEE Control Systems Magazine, vol. 10, no. 4, April 1990, pp. 3-5.

[113] K. J. Hunt, D. Sbarbaro, R. Zbikowski and P. J. Gawthrop, "Neural networks for control systems - a survey", Automatica, vol. 28, no. 6, Nov. 1992, pp. 1083-1112.

[114] P. J. Werbos, "An overview of neural networks for control", IEEE Control Systems Magazine, vol. 11, no. 1, Jan. 1991, pp. 40-41.

[115] H. Mori, K. Itou, H. Uematsu and S. Tsuzuki, "An artificial neural-net based method for predicting power system harmonics", IEEE Trans. on Power

Delivery, vol. 7, no. 1, Jan. 1992, pp. 402-409.

[116] Y. H. Pao and D. J. Sobajic, "Combined use of unsupervised and supervised learning for dynamic security assessment", IEEE Trans. on Power Systems, vol. 7, no. 2, May 1992, pp. 878-884.

[117] T. M. Peng, N. F. Hubele and G. G. Karady, "Advancement in the application of neural networks for short-term load forecasting", IEEE Trans. on Power Systems, vol. 7, no. 1, Feb. 1992, pp. 250-257.

[118] N. L. Santoso and O. T. Tan, "Neural-Net based real-time control of capacitors installed on distribution systems", IEEE Trans. on Power Delivery, vol. 5, no. 1, Jan. 1990, pp. 266-272.

[119] D. J. Sobajic and Y. H. Pao, "Artificial neural-net based dynamic security assessment for electric power systems", IEEE Trans. on Power Systems, vol. 4, no. 1, Feb. 1989, pp. 220-228.

[120] T. Hiyama, "Application of neural network to real time tuning of fuzzy logic PSS", Proceedings of the second international forum on applications of neural networks to power systems, 1993, pp 421-426.

[121] Y. Zhang, G. P. Chen, O. P. Malik and G. S. Hope, "A multi-input power system stabilizer based on artificial neural networks", Proceedings of IEEE WESCANEX-93, Saskatoon, Saskatchewan, Canada, May 17-18, 1993, pp. 240-246.

[122] M. L. Kothari, R. Segal and B. K. Ghodki, "Adaptive conventional power system stabilizer based on artificial neural network", International Conference on Power Electronics, Drives and Energy Systems for Industrial Growth, New Delhi, India, January 1996, pp. 1072-1077.

[123] M. K. Elsherbiny, G. Elsaady and E. A. Ibrahim, "Speed deviation driven adaptive neural network based power system stabilizer", Electric Power Systems Research, vol. 38, no. 3, Sept. 1996, pp. 169-175.

[124] Y. M. Park, S. H. Hyun and J. H. Lee, "A synchronous generator stabilizer design using neuro inverse controller and error reduction network", IEEE Trans. on Power Systems, vol. 11, no. 4, Nov. 1996, pp. 1969-1975.

[125] A. Hariri and O. P. Malik, "A fuzzy logic based power system stabilizer with learning ability", IEEE Trans. on Energy Conversion, vol. 11, no. 4, Dec. 1996, pp. 721-726.

[126] A. M. Sharaf and T. T. Lie, "An artificial neural network coordinated excitation/governor controller for synchronous generators", Electric Machines and Power Systems, vol. 25, no. 1, Jan. 1997, pp. 1-14.

[127] K. A. Elmetwally, N. D. Rao, O. P. Malik and G. Ramakrishna, "Application of a neural network as an integrated excitation controller", Electric Power Systems Research, vol. 42, no. 2, Aug. 1997, pp. 121-126.

[128] C. M. Lim and Q. Li, "An enhanced adaptive neural network control scheme for power systems", IEEE Trans. on Energy Conversion, vol. 12, no. 2, June 1997, pp. 166-172.

[129] S. Pillutla and S. Keyhani, "Power system stabilization based on modular neural network architecture", International Journal of Electrical Power and Energy Systems, vol. 19, no. 6, Aug. 1997, pp. 411-418.

[130] S. J. Cheng, J. R. Zhou and L. Guan, "An on-line self-learning power system stabilizer using a neural network method", IEEE Trans. on Power Systems, vol. 12, no. 2, May 1997, pp. 926-931.

[131] E. H. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller", Int. J. Man-Machine Studies, vol. 7, no. 1, 1975, pp. 1-13.

[132] J. P. LaSalle and S. Lefschetz, "Stability by Liapunov's direct method with applications", New York: Academic, 1961.

[133] M. Sugeno, "Industrial applications of fuzzy control", Amsterdam: North Holland, 1985.

[134] P. J. King and E. H. Mamdani, "The application of fuzzy control systems to industrial processes", Automatica, vol. 13, no. 3, 1977, pp. 235-242.

[135] E. H. Mamdani, "Applications of fuzzy algorithms for simple dynamic plant", Proceedings of IEE, vol. 121, no. 12, 1974, pp. 1585-1588.

[136] E. H. Mamdani, "Advances in the linguistic synthesis of fuzzy controllers", Int. J. Man-Machine Studies, vol. 8, no. 6, 1976, pp. 669-678.

[137] E. H. Mamdani, "Application of fuzzy logic to approximate reasoning using linguistic synthesis", IEEE Trans. Computer, vol. C-26, no. 12, 1977, pp. 1182-1191.

[138] E. H. Mamdani, J. J. Ostergaard and E. Lembessis, "Use of fuzzy logic for

implementing rule-based control of industrial processes", Fuzzy Sets and Decision Analysis, North Holland: Amsterdam, 1984, pp 429-445.

[139] L. A. Zadeh, "Fuzzy algorithm", Information Control, vol. 12, 1968, pp. 94-102.

[140] L. A. Zadeh, "Toward a theory of fuzzy systems", in Aspects of Network and System Theory, R. E. Kalman and N. DeClaris, Ed. New York: Holt, Rinehart and Winston, 1971, pp. 469-490.

[141] L. A. Zadeh, "A rationale for fuzzy control", Trans. ASME, J. Dynam. Syst. Measur. Control, vol. 94, 1972, pp. 3-4.

[142] L. A. Zadeh, "Outline of a new approach to the analyais complex systems and decision processes", IEEE Trans. Syst. Man Cybern., vol. 3, 1973, pp. 28-44.

[143] O. Yagishita, O. Itoh and M. Sugeno, "Application of fuzzy reasoning to the water purification process", in Industrial applications of fuzzy control, M. Sugeno, Ed. Amsterdam: North-Holland, 1985, pp. 19-40.

[144] S. Yasunobu and S. Miyamoto, "Automatic train operation by predictive fuzzy control", in Industrial applications of fuzzy control, M. Sugeno, Ed. Amsterdam: North-Holland, 1985, pp. 1-18.

[145] S. Yasunobu and T. Hasegawa, "Evaluation of an automatic container crane operation system based on predictive fuzzy control", Control Theory Adv. Technol., vol. 2, no. 3, 1986, pp. 419-432.

[146] J. A. Bernard, "Use of rule-based system for process control", IEEE Control Systems Magazine, vol. 8, no. 5, 1988, pp. 3-13.

[147] T. Yamakawa, "High speed fuzzy controller hardware system", in proc. 2nd Fuzzy System Symp., Japan, 1986, pp. 122-130.

[148] M. Togai and S. Chiu, "A fuzzy accelerator and a programming environment for real-time fuzzy control", in Proc. 2nd IFSA Congress, Tokyo, Japan, July 1987, pp. 147-151.

[149] M. Togai and H. Watanabe, "Expert system on a chip: an engine for real-time approximate reasoning", IEEE Expert Systems Magazine, vol. 1, 1986, pp. 55-62.

[150] T. Yamakawa and T. Miki, "The current mode fuzzy logic integrated circuits fabricated by the standard CMOS process", IEEE Trans. Computer, vol. 35, no.

2, 1986, pp. 161-167.

[151] T. Yamakawa and K. Sasaki, "Fuzzy memory device", in Proc. 2nd IFSA Congress, Tokyo, Japan, July 1987, pp. 551-555.

[152] T. Yamakawa, "A simple fuzzy computer hardware system employing min and max operations -A challenge to 6th generation computer", in Proc. 2nd IFSA Congress, Tokyo, Japan, July 1987.

[153] T. Yamakawa, "Fuzzy microprocessors -Rule chip and defuzzifier chip", in Int. Workshop on Fuzzy System Applications, Iizuka, Japan, Aug. 1988, pp. 51-52.

[154] S. Korner, "Laws of Thought", Encyclopedia of Philosophy, vol. 4, MacMillan, NY, 1967, pp. 414-417.

[155] B. Kosko, "Fuzzy thinking", Prentice-Hall, Englewood Cliffs, NJ, 1993.

[156] M. Black, "Vagueness: an exercise in logical analysis", Philosophy of Science, vol. 4, 1937, pp. 427-455.

[157] L. A. Zadeh, "From circuit theory to systems theory", Proceedings of Institution of Radio Engineers, vol. 50, 1962, pp. 856-865.

[158] D. Mcheill and P. Freiberg, "Fuzzy logic: the discovery of a revolutionary computer technology and how it is changing our world", Simon & Schuster, 1993.

[159] G. J. Klir and B. Yuan, "Fuzzy sets and fuzzy logic, theory and applications", Prentice-Hall, Upper Saddle River, New Jersey, USA, 1995.

[160] T. Kuhn, "The structure of scientific revolutions", Univ. of Chicago Press, 1962.

[161] B. Kosko, "Neural networks and fuzzy systems, a dynamical systems approach to machine intelligence", Prentice-Hall, Englewood Cliffs, NJ, 1992.

[162] Y. Murayama and T. Terano, "Optimizing control of diesel engine", in Industrial Applications of Fuzzy Control, M. Sugeno, Ed., Amsterdam: North-Holland, 1985, pp. 105-124.

[163] H. Hellendoorn and C. Thomas, "Defuzzification in fuzzy controllers", Journal of Intelligent Systems, vol. 1, 1993, pp. 109-123.

[164] B. Kosko and J. A. Dickerson, "Function approximation with additive fuzzy systems", in Theoretical Aspects of Fuzzy Control, H. T. Nguyen, M. Sugeno,

R. Tong and R. Yager, Editors, Wiley, New York, 1994, chapter 12.

[165] P. Pacini and B. Kosko, "Comparison of fuzzy and Kalman filter target-tracking control systems", in Neural Networks and Fuzzy Systems, Prentice-Hall, Englewood Cliffs, NJ, pp. 379-406.

[166] L. X. Wang and J. M. Mendel, "Fuzzy basis functions, universal approximation,and orthogonal least squares learning", IEEE Trans. on Neural Networks, vol. 3, Sept. 1992, pp. 807-813.

[167] H. M. Kim and J. M. Mendel, "Fuzzy basis functions: Comparisons with other basis functions", IEEE Trans. on Fuzzy Systems, vol. 3, no. 2, May 1995, pp. 158-168.

[168] T. Poggio and F. Girosi, "Networks for approximation and learning", Proceedings of IEEE, vol. 78, Sept. 1990, pp. 1481-1497.

[169] D. F. Specht, "A general regression neural network", IEEE Trans. on Neural Networks, vol. 2, 1991, pp. 568-576.

[170] D. L. Alspach and H. W. Sorenson, "Nonlinear Bayesian estimation using Gaussian sum approximations", IEEE Trans. on Automatic Control, vol. 17, Aug. 1972, pp. 439-448.

[171] A. Kalam, "Cogeneration, a short course", Victoria University of Technology, Melbourne, Australia, 1994.

[172] W. G. Heffron and R. A. Philips, "Effects of modern amplidyne voltage regulator in underexcited operating of large turbine generators", AIEE Transactions, vol. PAS-71, August 1952, pp. 692-697.

[173] J. J. D'Azzo, C. H. Houpis, "Feedback control system analysis and synthesis", Second Edition, McGraw-Hill, Inc., New York, USA, 1966.

[174] T. Hiyama, "Real time control of micro-machine system using micro-computer based fuzzy logic power system stabilizer", IEEE Trans. on Energy Conversion, vol. 9, no. 4, Dec. 1994, pp 724-731.

[175] M. A. M. Hassan and O.P. Malik, "Laboratory evaluation and test results for rule-based power system stabilisers: a comparative study", Intelligent Systems Engineering, Spring 1993, pp. 52-60.

[176] T. Hiyama, S. Oniki, and H. Nagashima, "Experimental studies on micro-computer based fuzzy logic power system stabilizer", Proceedings of the

Second International Forum on Applications of Neural Networks to Power Systems, 1993, pp. 212-217.

[177] N. Hosseinzadeh, A. Kalam and W. S. Lee, "A fuzzy logic based power system stabiliser for a synchronous generator", eecon'95-Electrical Energy Conference, Adelaide, Australia, Sept. 1995, pp 176-181.

[178] R. R. Yager and D. P. Filev, "Essentials of fuzzy modelling and control", John Wiley & Sons, 1994.

[179] T. Hiyama, "Rule-Based stabilizer for multi-machine power systems", IEEE Trans. on Power Systems, vol. 5, no. 2, 1990, pp. 403-411.

[180] H. Berenji, "Neural networks and fuzzy logic in intelligent control", Proceedings of the 5th IEEE International Symposium on Intelligent Control, Philadelphia, USA, Sept. 1990, pp 916-920.

[181] J.-S. R. Jang and C.-T. Sun, "Neuro-Fuzzy modelling and control", Proceedings of the IEEE, vol. 83, no. 3, March 1995, pp 378-406.

[182] D. E. Rumelhart, G.E. Hinton and R.J. Williams, "Learning internal representations by error propagation", Parallel Distributed Processing, vol. 1, MIT Press, 1986, pp 318-362.

[183] A. M. Sharaf and T. T. Lie, "A hybrid neuro-fuzzy power system stabilizer", IEEE International Conference on Neural Networks, vol. 3, New York, 1994, pp 1760-1765

[184] J. M. Zurada, "Introduction to artificial neural systems", West Publishing Company, 1992.

[185] H. Demuth and M. Beale, "Neural network toolbox for use with MATLAB", MathWorks Inc., 1994, pp 5/33-5/34.

[186] G. Cybenko, "Approximations by superpositions of a sigmoidal function", Mathematics of Signals and Systems, vol. 2, 1989, pp. 303-314.

[187] K. Funahashi, "On the approximate realisation of continuous mappings by neural networks", IEEE Trans. on Neural Networks, vol. 2, 1989, pp. 183-192.

[188] K.S. Narendra and K. Parthasarathy, "Identification and control of dynamic systems using neural networks," IEEE Trans. on Neural Networks, vol. 1, no. 1, 1990, pp. 4-27.

[189] K.S. Narendra and K. Parthasarathy, "Neural networks and dynamical systems", International Journal of Approximate Reasoning, vol. 6, 1992, pp. 109-131.

[190] A. S. Levin, and K.S. Narendra, "Control of nonlinear dynamical systems using neural networks - part II: observability, identification, and control", IEEE Transactions on Neural Networks, vol. 7, no. 1, January 1996, pp. 30-42.

[191] F. C. Chen, "Backpropagation neural networks for nonlinear self-tuning adaptive control", IEEE Control Systems Magazine, vol. 10, 1990, pp. 44-48.

[192] D. Nguyen and B. Widrow, "The truck backer-upper: an example of self-learning in neural networks", Proc. International Joint Conference on Neural Networks, Washington, D.C., vol. 2, June 1989, pp. 11357-11363.

[193] M. I. Jordan and R. A. Jacobs, "Learning to control an unstable system with forward modelling", Advances in Neural Information Processing Systems, Morgan Kaufman, San Mateo, CA., vol. 2, 1990, pp. 324-331.

[194] L. X. Wang, "A course in fuzzy systems and control", Prentice-Hall, Upper Saddle River, NJ, 1997.

[195] L. X. Wang, "Stable adaptive fuzzy control of nonlinear systems", IEEE Trans. on Fuzzy Systems, vol. 1, no. 2, May 1993, pp. 146-155.

[196] R. M. Sanner and J. E. Slotine, "Gaussian networks for direct adaptive control", in Proc. American Control Conf., 1991, pp. 2153-2159.

[197] L. X. Wang, "Stable adaptive fuzzy controllers with application to inverted pendulum tracking", IEEE Trans. on Systems, Man and Cybernetics, Part B, vol. 26, no. 5, Oct. 1996, pp. 677-691.

[198] J. E. Slotine and W. Li, "Applied nonlinear control", Prentice-Hall, Englewood Cliffs: NJ, 1991.

[199] A. Isidori, "Nonlinear control systems", Springer-Verlag: Berlin, 1989.

[200] D. G. Luenberger, "Linear and nonlinear programming", Addison-Wesley Publishing Company: Reading, MA:USA, 1984.

Appendices

APPENDIX A

THE PARAMETERS OF THE INDUSTRIAL COGENERATOR USED IN THE SIMULATION STUDIES:

The parameters of the synchronous machine, excitation system, and conventional PSS are as follows:

a) Synchronous machine constants:

X_d = 2.64 pu X_d' = 0.28 pu

X_d'' = 0.2 pu X_q = 1.32 pu

X_q' = 0.29 pu X_q'' = 0.21 pu

X_l = 0.1 pu R_a = 0.003 pu

T_{d0}' = 5.79 sec T_{d0}'' = 0.044 sec

T_{q0}' = 1.2 sec T_{q0}'' = 0.084 sec

H = 3.8 sec K_D = 0.01

f = 50 Hz

b) Excitation system constants:

K_A = 100 T_A = 0.05 T_R = 0.015

$E_{f,max}$ = 5.0 $E_{f,min}$ = –2.0

c) PSS saturation levels:

$V_{s,max}$ = 0.2 $V_{s,min}$ = –0.2

APPENDIX B

The following set of training data was obtained by performing various tests on the power system. In each case the scaling factors K_p and K_d were adjusted such that the performance index J_p was minimised. This data has been used to train an artificial neural network which has been exploited to tune the fuzzy logic PSS on-line.

Table B.1. The set of training data to train the neural network tuner.

P_o	Q_o	X_e	K_p	K_d	J_p
0.4	-0.4	0.1	0.13	0.38	20.3
0.4	0	0.1	0.14	0.59	17.2
0.4	0.4	0.1	0.16	0.78	13.6
0.4	0.8	0.1	0.19	1.9	9.5
0.4	-0.4	0.2	0.36	0.38	39.7
0.4	0	0.2	0.36	0.59	20.6
0.4	0.4	0.2	0.38	0.78	17.9
0.4	0.8	0.2	0.41	1.9	13.2
0.4	-0.4	0.3	0.36	0.38	62.5
0.4	0	0.3	0.36	0.46	29.7
0.4	0.4	0.3	0.38	0.5	26.9
0.4	0.8	0.3	0.45	2.5	18.8
0.4	-0.4	0.4	0.37	0.68	112.7
0.4	0	0.4	0.39	0.8	52.4
0.4	0.4	0.4	0.44	2.05	47.3
0.4	0.8	0.4	0.49	2.6	29.8
0.6	-0.4	0.1	0.19	0.36	22.8

Table B.1. The set of training data to train the neural network tuner.

0.6	0	0.1	0.19	0.5	23.7
0.6	0.4	0.1	0.19	1.6	18.7
0.6	0.8	0.1	0.19	2.65	12.0
0.6	-0.4	0.2	0.35	0.9	51.8
0.6	0	0.2	0.38	0.9	37.0
0.6	0.4	0.2	0.38	1.61	27.6
0.6	0.8	0.2	0.41	2.2	17.8
0.6	-0.4	0.3	0.37	0.81	114.8
0.6	0	0.3	0.48	1.12	40.9
0.6	0.4	0.3	0.55	1.25	36.8
0.6	0.8	0.3	0.55	2.65	27.2
0.6	-0.3	0.4	0.55	0.82	347.8
0.6	0	0.4	0.55	1.18	54.5
0.6	0.4	0.4	0.69	1.5	42.1
0.6	0.8	0.4	0.7	3.3	37.0
0.8	-0.4	0.1	0.28	0.78	27.8
0.8	0	0.1	0.28	1.2	28.7
0.8	0.4	0.1	0.27	1.6	19.8
0.8	0.8	0.1	0.27	2.7	17.2
0.8	-0.4	0.2	0.29	0.85	82.1
0.8	0	0.2	0.35	0.93	40.1
0.8	0.4	0.2	0.54	2.1	26.3
0.8	0.8	0.2	0.54	3.2	27.0
0.8	-0.3	0.3	0.4	0.8	516.5
0.8	0	0.3	0.44	1.1	65.7
0.8	0.4	0.3	0.55	1.8	42.8
0.8	0.8	0.3	0.56	3.3	40.3

Table B.1. The set of training data to train the neural network tuner.

0.8	0	0.4	0.64	1.7	206.9
0.8	0.4	0.4	0.64	2.0	68.7
0.8	0.8	0.4	0.64	3.1	57.6
1.0	-0.4	0.1	0.32	0.92	45.1
1.0	0	0.1	0.34	0.94	24.7
1.0	0.4	0.1	0.34	1.8	21.0
1.0	0.8	0.1	0.32	2.4	27.4
1.0	-0.4	0.2	0.38	0.8	363.8
1.0	0	0.2	0.38	1.05	55.3
1.0	0.4	0.2	0.38	1.45	38.4
1.0	0.8	0.2	0.34	2.0	43.1
1.0	0	0.3	0.57	1.3	315.6
1.0	0.4	0.3	0.5	1.9	73.8
1.0	0.8	0.3	0.5	2.6	71.5
1.0	0.4	0.4	0.85	1.9	286.8
1.0	0.8	0.4	0.72	2.45	121.3

APPENDIX C

The parameters of the synchronous machine used to make a comparison between adaptive fuzzy logic PSSs and other PSSs:

X_d = 2.2 pu X_d' = 0.22 pu

X_d'' = 0.2 pu X_q = 1.01 pu

X_q' = 0.24 pu X_q'' = 0.21 pu

X_l = 0.05 pu R_a = 0.002 pu

T_{d0}' = 4.5 sec T_{d0}'' = 0.05 sec

T_{q0}' = 1.3 sec T_{q0}'' = 0.04 sec

H = 5 sec K_D = 0.01

f = 50 Hz

The same excitation system and PSS are used as introduced in Appendix A.

APPENDIX D

SOURCE CODES

The software developed in this research is written in the MATLAB code. In this appendix, the main programs are presented.

Part I: Programs related to steady state stability

% freqDesign.m %

```
% conventional design using frequency response
% figures 4.5, 4.7, 4.8 in the thesis are obtained by running this program
clear
% define the plant
gendefn;
APPROX = 0;
if APPROX == 1 %% LOOP1
    TRsmall = 0
    if TRsmall == 1  %% LOOP2
        OMEGAx = sqrt(K6*KA/(Td0p*Ta))
        ZETAx = (Ta+T3)/(2*OMEGAx*T3*Ta);
        num_Gx = [K2*KA/(Td0p*Ta)]; den_Gx = [1 2*ZETAx*OMEGAx OMEGAx^2];
    else
        num_Gx = K2*K3*KA*[TR 1]; den_Gx = [TR*Ta*T3 TR*Ta+TR*T3+Ta*T3 TR+Ta+T3 1+K3*KA*K6];
    end   %% End of LOOP2

    num_Gy = [-1 0]; den_Gy = [2*H Kd w0*K1];
    [nGo,dGo] = series(num_Gx,den_Gx,num_Gy,den_Gy)
else
    num_GG1 = KA*K3; den_GG1 = [Ta*T3 Ta+T3 1];
    num_GG2 = -[K2*K3 K2 0]; den_GG2 = [2*H*T3 2*H+Kd*T3 K1*T3*w0+Kd (K1-K2*K3*K4)*w0];
    [num_GG,den_GG] = series(num_GG1,den_GG1,num_GG2,den_GG2);
    num_HH1 = [-2*H*K6 -K6*Kd (K2*K5-K6*K1)*w0]; den_HH1 = [0 K2 0];
    num_HH2 = 1; den_HH2 = [TR 1];
    [num_HH,den_HH] = series(num_HH1,den_HH1,num_HH2,den_HH2);
    [nGo,dGo] = feedback(num_GG,den_GG,num_HH,den_HH,-1);
end   % End of LOOP1

% Transfer function of 1/Gp(s)
nIH = dH, dIH = nH
[nGoh,dGoh] = series(nGo,dGo,nH,dH);
% Gu(jw) = G1(jw)/[1-G1(jw)]
nGuh = nGoh; dGuh = dGoh-nGoh;

% Bode plots
```

```
w = [.1 : 0.02 : 10, 10 : 0.1 :100];
LW = 1.6;  % Line Width of the curves
% bode plot of Go(jw)
figure(1);clf;
bode(nGo,dGo,w);
title('Frequency response of G(jw)','FontSize',14);
xlabel('Frequency (rad/sec)','FontSize',14); ylabel('Phase (degrees)        Magnitude (dB)','FontSize',14);
hh = get(gcf,'children'); hhh = get(hh(1),'Children'); set(hhh,'LineWidth',LW);
hhh = get(hh(2),'Children'); set(hhh,'LineWidth',LW);

% bode plot of H(jw)
figure(2);clf;
bode(nH,dH,w);
%subplot(211);
title('frequency response of the CPSS','FontSize',14);
xlabel('Frequency (rad/sec)','FontSize',14); ylabel('Phase (degrees)        Magnitude (dB)','FontSize',14);
hh = get(gcf,'children'); hhh = get(hh(1),'Children'); set(hhh,'LineWidth',LW);
hhh = get(hh(2),'Children'); set(hhh,'LineWidth',LW);

% bode (magnitude) plot of Go(jw) and 1/Gp(jw)
figure(3);clf;
mybode(nGo,dGo,w); hold on; mybode(nIH,dIH,w);
%subplot(211);
title('frequency response of G(jw) and 1/Gp(jw)','FontSize',14);
xlabel('Frequency (rad/sec)','FontSize',14); ylabel('Magnitude (dB)','FontSize',14);
hh = get(gca,'children'); set(hh,'LineWidth',LW); set(hh(2),'LineStyle','--');
hh = get(gcf,'children'); set(hh(1),'FontSize',14);

% bode plot of Go(jw)*H(jw)
figure(4);clf;
bode(nGoh,dGoh,w);
%subplot(211);
title('frequency response of G(jw)*Gp(jw)','FontSize',14);
xlabel('Frequency (rad/sec)','FontSize',14); ylabel('Phase (degrees)        Magnitude (dB)','FontSize',14);
hh = get(gcf,'children'); hhh = get(hh(1),'Children'); set(hhh,'LineWidth',LW);
hhh = get(hh(2),'Children'); set(hhh,'LineWidth',LW);

% bode plot of Gu(jw)
figure(5);clf;
bode(nGuh,dGuh,w);
%subplot(211);
title('frequency response of Gu(jw)','FontSize',14);
xlabel('Frequency (rad/sec)','FontSize',14); ylabel('Phase (degrees)        Magnitude (dB)','FontSize',14);
hh = get(gcf,'children'); hhh = get(hh(1),'Children'); set(hhh,'LineWidth',LW);
hhh = get(hh(2),'Children'); set(hhh,'LineWidth',LW);

% bode plot of G1(jw)=Go(jw)/[1-Go(jw)*Gp(jw)]
[nG1,dG1] = series(dH,nH,nGuh,dGuh);
figure(6);clf;
bode(nG1,dG1,w);
%subplot(211);
```

```
title('closed loop frequency response','FontSize',14);
xlabel('Frequency (rad/sec)','FontSize',14); ylabel('Phase (degrees)        Magnitude (dB)','FontSize',14);
hh = get(gcf,'children'); hhh = get(hh(1),'Children'); set(hhh,'LineWidth',LW);
hhh = get(hh(2),'Children'); set(hhh,'LineWidth',LW);
```

% NOPSS_dist.m %

```
% plot results from simulation of the system without PSS after a disturbance occurs

clear, tic
gendefn;
[t,x,y] = sim('genAVR3',Tfinal); deltaw = y(:,1); SPdevw = y(:,2); Vtw = y(:,3); ind = y(:,4);
NS = Tfinal/max_st;
J = 0;
for I = 1 : NS; J = J + ind(I); end
J = J
toc
LW = 1.6;  % Line width for the curves
figure(1);clf;
plot(t,SPdevw);
xlabel('t, sec','FontSize',14); ylabel('speed deviation, rad/sec','FontSize',14);
hh = get(gca,'children'); set(hh,'LineWidth',LW); hh = get(gcf,'children'); set(hh(1),'FontSize',14);

figure(2);clf;
plot(t,deltaw*180/pi);
xlabel('t, sec','FontSize',14); ylabel('Rotor angle, degrees','FontSize',14)
hh = get(gca,'children'); set(hh,'LineWidth',LW); hh = get(gcf,'children'); set(hh(1),'FontSize',14);

figure(3);clf;
plot(t,Vtw);
xlabel('t, sec','FontSize',14); ylabel('Change in terminal voltage, pu','FontSize',14)
hh = get(gca,'children'); set(hh,'LineWidth',LW); hh = get(gcf,'children'); set(hh(1),'FontSize',14);
```

% gendefn.m %

```
% generator model parameters

param
% Sampling period
Ts = 0.005;
Tfinal = 5;
rel_err = 0; min_st = 0.005; max_st = 0.005;
Af = 6;  % Acceleration scaling factor for Hiyama-Type FPSS
Efd0 = 0.95   % Adjusting Efd0 manually
```

```
% param.m
% calculate parameters of the generator from given data

% Transformer parameters
Rtr = 0.005; Xtr = 0.05;
```

```
% Transmission line parameters
Re = 0.03; Xe = 0.15
% Total line impedance
Rt = Re+Rtr; Xt = Xe+Xtr;
% Operating point
Pt0 = 0.9, Fp = 0.9, Vinf = 1.0;
LAG = 1          % Lagging phase
Phi=sign(LAG)*acos(Fp);
Qt0 = Pt0*sin(Phi)/Fp
% saturation coefficients
Ksd = 0.835; Ksq = 0.835;

Latrobe = 1;
if Latrobe == 1
    Xd=2.64; Xdp=0.28; Xddp=0.2; Xq=1.32; Xqp=0.29; Xqdp=0.21; Xl=0.1; Ra=0.003;
    Td0p=5.79; Td0dp=0.044; Tq0p=1.2; Tq0dp=0.084; H = 3.8; Kd = 0.01;
else
    % Crown cogenerator standard parameters
    Xd=2.1877; Xdp=0.1763; Xddp=0.1384; Xq=1.0152; Xqp=0.18; Xqdp=0.1664; Xl=0.1; Ra=0.003;
    Td0p=4.14743; Td0dp=0.01483; Tq0p=0.6; Tq0dp=0.01387; H = 5; Kd = 0.02;
end

f=50;w0=2*pi*f;

% AVR parameters
KA = 100; Ta = 0.05;

% Terminal voltage transducer parameter
TR = 0.015;

% PSS saturation levels
vsmin = -0.2; vsmax = 0.2;

% define the CPSS
Kstab = 20;T0 = 10; T1z = 0.12; T1p = 0.014; T2z = 0.12; T2p = 0.014;

num_WW = [Kstab*T0 0]; den_WW = [T0 1];
num_Lead1 = [T1z 1]; den_Lead1 = [T1p 1];
num_Lead2 = [T2z 1]; den_Lead2 = [T2p 1];
[num_Lead,den_Lead] = series(num_Lead1,den_Lead1,num_Lead2,den_Lead2);
[nH,dH] = series(num_WW,den_WW,num_Lead,den_Lead);

figure(1); clf
bode(num_WW,den_WW)
figure(2);clf
bode(num_Lead1,den_Lead1)
figure(3);clf
bode(num_Lead2,den_Lead2)

% Calculate fundamental parameters
Xad = Xd - Xl; Xaq = Xq - Xl;
```

```
Xads = Ksd*Xad; Xaqs = Ksq*Xaq;
Xds = Xads + Xl; Xqs = Xaqs + Xl;
Xfd = Xad*(Xdp-Xl)/(Xad+Xl-Xdp);
X1q = Xaq*(Xqp-Xl)/(Xaq+Xl-Xqp); X1d = Xad*Xfd*(Xl-Xddp)/((Xad+Xfd)*(Xddp-Xl)-Xad*Xfd);
X2q = Xaq*X1q*(Xl-Xqdp)/((Xaq+X1q)*(Xqdp-Xl)-Xaq*X1q);
Rfd = (Xad+Xfd)/(Td0p*w0);
R1d = (X1d+Xad*Xfd/(Xad+Xfd))/(Td0dp*w0); R1q = (Xaq+X1q)/(Tq0p*w0);
R2q = (X2q+Xaq*X1q/(Xaq+X1q))/(Tq0dp*w0);
Xffd = Xads+Xfd;

% Operating point calculations
Pt0loss = (1.0^2)*Rt; Ir = 1.0 - Pt0loss;
tanPhi = tan(Phi);
a = Rt*tanPhi - Xt; b = Vinf; c = tanPhi*(Ir*Vinf+Rt*Ir^2)-Xt*Ir^2;
Ix = (-b+sqrt(b^2-4*a*c))/(2*a);
Vacomp = (Vinf-Xt*Ix+Rt*Ir) + j*(Xt*Ir+Rt*Ix); Va = abs(Vacomp)

Ia = Pt0/(Va*Fp); PHAY = Phi;
Ir = Ia*Fp; Ix = -Ia*sin(PHAY);
delta = atan(((Xq+Xt)*Ir+(Ra+Rt)*Ix)/(Vinf-(Xq+Xt)*Ix+(Ra+Rt)*Ir))
beta = atan((Xt*Ir+Rt*Ix)/(Vinf-Xt*Ix+Rt*Ir));
delMbet = delta - beta;
Vd0 = -Va*sin(delMbet); Vq0 = Va*cos(delMbet);
Id0 = -Ia*sin(delMbet + PHAY); Iq0 = Ia*cos(delMbet + PHAY);

sqrt3 = sqrt(3); eq0 = sqrt3*Vq0; iq0 = sqrt3*Iq0; id0 = sqrt3*Id0;
ifd0=(eq0+Ra*iq0+Xds*id0)/Xads;
SYfd0=(Xads+Xfd)*ifd0-Xads*id0;

% K parameter calculations

% Infinite bus voltage calculations
delMalph = delta;       % alpha = 0 because the infinite bus voltage is the reference
sindMa = sin(delMalph); cosdMa = cos(delMalph);

% parameters calculations
E0 = Vq0 + Ra*Iq0 - Xd*Id0; Efd0 = E0;
Eqa0 = E0 + (Xd - Xq)*Id0;
KI = 1/(Rt^2 + (Xq+Xt)*(Xdp+Xt));
K11 = Eqa0*(Rt*sindMa+(Xdp+Xt)*cosdMa); K12 = Iq0*(Xq-Xdp)*((Xq+Xt)*sindMa - Rt*cosdMa);
K1 = KI * Vinf * (K11 + K12)
K2 = KI*(Iq0*(Rt^2+(Xq+Xt)^2) + Eqa0*Rt)
K3 = 1/(1 + KI*(Xd-Xdp)*(Xq+Xt))
K4 = Vinf*KI*(Xd-Xdp)*((Xq+Xt)*sindMa - Rt*cosdMa)
K51 = (KI*Vinf*Xdp*Vq0/Va)*(Rt*cosdMa - (Xq+Xt)*sindMa);
K52 = (KI*Vinf*Xq*Vd0/Va)*((Xdp+Xt)*cosdMa + Rt*sindMa);
K5 = K51 - K52
K6 = (Vq0/Va)* (1 - KI*Xdp*(Xq+Xt)) - (Vd0/Va)*KI*Xq*Rt
T3 = K3*Td0p*Xad/Xffd
```

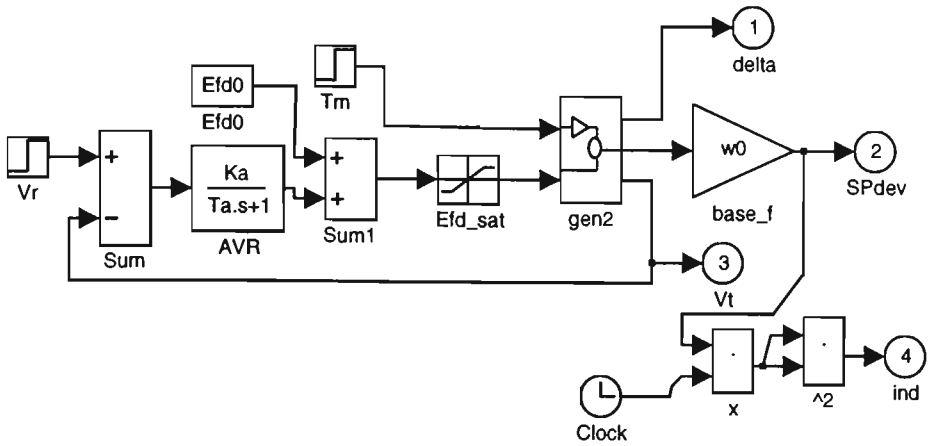


Fig. H.1. Block diagram of genAVR3 to be used with SIMULINK

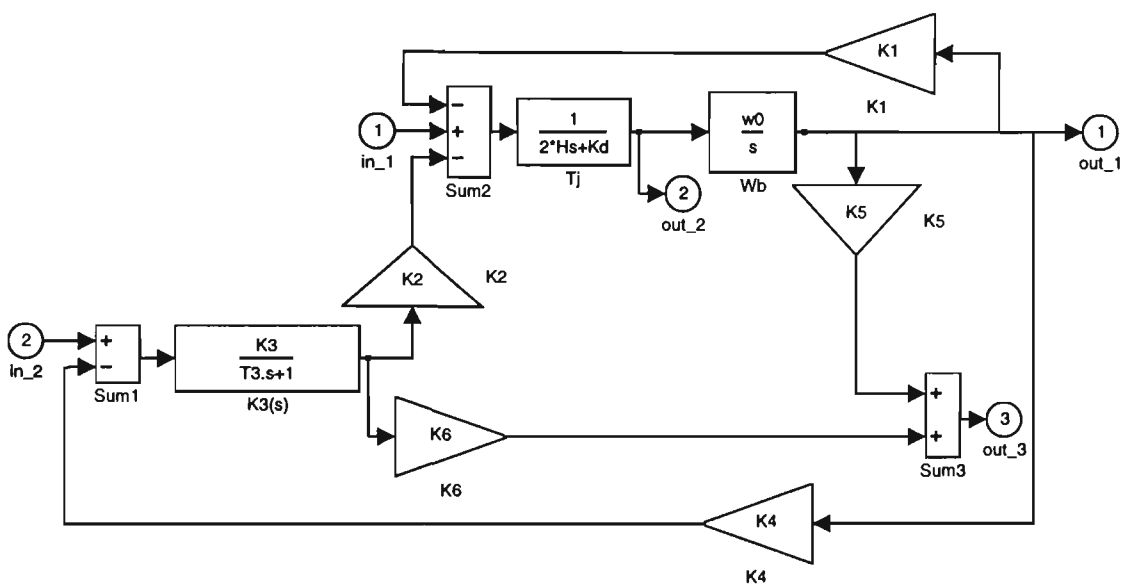


Fig. H.2. Block diagram of gen2 subsystem

% conv_dist.m %

% plot results from simulation of the system with conventional PSS after a disturbance occurs in the system

```
clear
tic
gendefn;
[t,x,y] = sim('convPSScog',Tfinal);
delta = y(:,1); spdev = y(:,2); vpss = y(:,5); Vt = y(:,3); ind = y(:,4);
NS = Tfinal/max_st;
Jc = 0;
for I = 1 : NS; Jc = Jc + ind(I); end
Jc = Jc
```

```
toc
LW = 1.6;  % Line Width for the curves
figure(1);clf;
plot(t,spdev);
title('speed deviation using conventional PSS (case 1)')
xlabel('t, sec','FontSize',14); ylabel('speed deviation, rad/sec','FontSize',14); grid
hh = get(gca,'children'); set(hh,'LineWidth',LW); hh = get(gcf,'children'); set(hh(1),'FontSize',14);

figure(2);clf;
plot(t,delta*180/pi);
xlabel('t, sec','FontSize',14); ylabel('Rotor angle, degrees','FontSize',14); grid
hh = get(gca,'children'); set(hh,'LineWidth',LW); hh = get(gcf,'children'); set(hh(1),'FontSize',14);

figure(3);clf;
plot(t,vpss);
title('output of conventional PSS (case 1)')
xlabel('t, sec','FontSize',14); ylabel('Stabilising signal, pu','FontSize',14); grid
hh = get(gca,'children'); set(hh,'LineWidth',LW); hh = get(gcf,'children'); set(hh(1),'FontSize',14);

figure(4);clf;
plot(t,Vt);
title('terminal voltage with conventional PSS (case 1)')
xlabel('t, sec','FontSize',14); ylabel('terminal voltage, pu','FontSize',14); grid
hh = get(gca,'children'); set(hh,'LineWidth',LW); hh = get(gcf,'children'); set(hh(1),'FontSize',14);
```

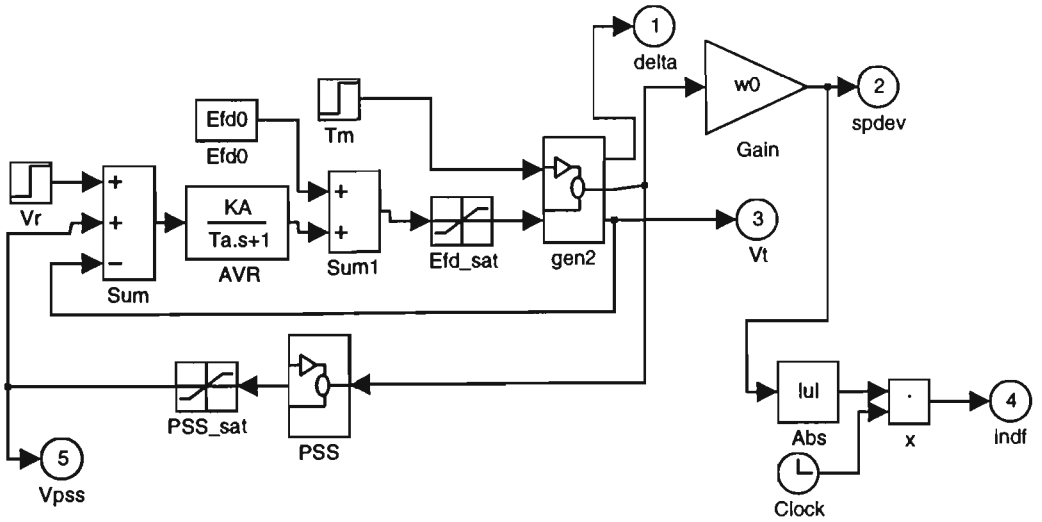


Fig. H.3. Block diagram of convPSScog to be used with SIMULINK

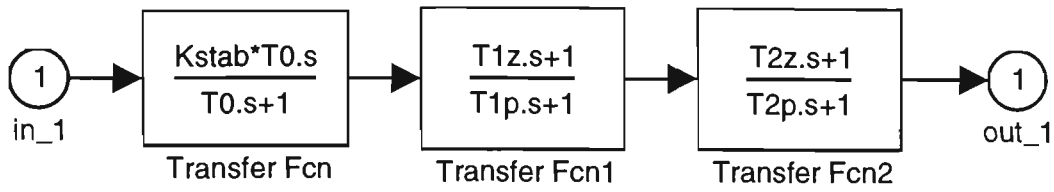


Fig. H.4. Block diagram of PSS subsystem

% FPSS_dist.m %

% plot results from simulation of the system with rule-based FPSS after a disturbance occurs

```
clear
tic
gendefn;
global spCOEF accCOEF uCOEF cont_FUZmat

% calling Mamdani rule-based fuzzy controller
cont_FUZmat = readfis('/pgr/nasser/matlab/FLPSS/wcff_PSS/mamdn');
spCOEF = 0.15; accCOEF = 0.9; uCOEF = 0.5;          %Optimum for O.P. Pt0=0.9, PF=0.9, Xe=0.15

[t,x,y] = sim('FPSS_Cog',Tfinal);
delta = y(:,1); spdev = y(:,2); vpss = y(:,5); Vt = y(:,3); ind = y(:,4);
NS = Tfinal/max_st;

JRf = 0;
for I = 1 : NS; JRf = JRf + ind(I); end
JRf = JRf
toc

figure(1);clf;
plot(t,spdev);
title('speed deviation using conventional PSS (case 1)')
xlabel('t, sec'); ylabel('speed deviation, rad/sec'); grid

figure(2);clf;
plot(t,delta*180/pi);
xlabel('t, sec'); ylabel('Rotor angle, degrees'); grid

figure(3);clf;
plot(t,vpss);
title('output of conventional PSS (case 1)')
xlabel('t, sec');ylabel('Stabilising signal, pu'); grid

figure(4);clf;
plot(t,Vt);
title('terminal voltage with conventional PSS (case 1)')
xlabel('t, sec'); ylabel('terminal voltage, pu'); grid
```

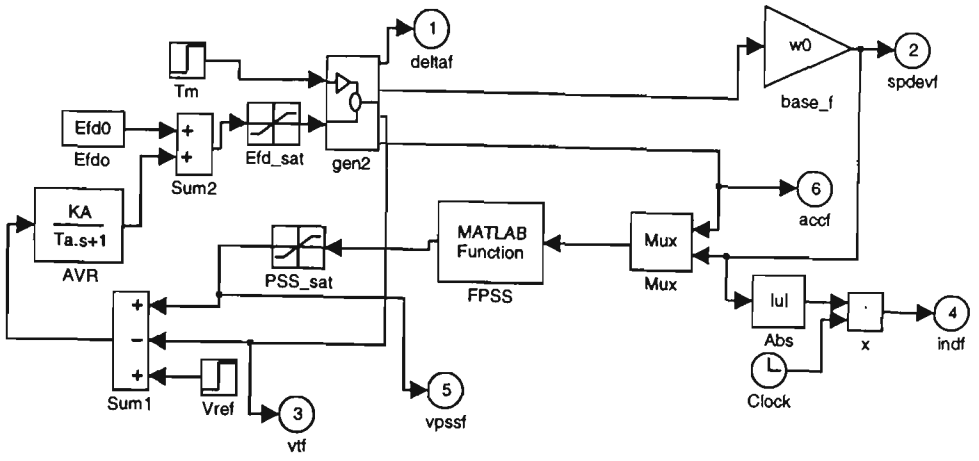


Fig. H.5. Block diagram of FPSS_Cog to be used with SIMULINK

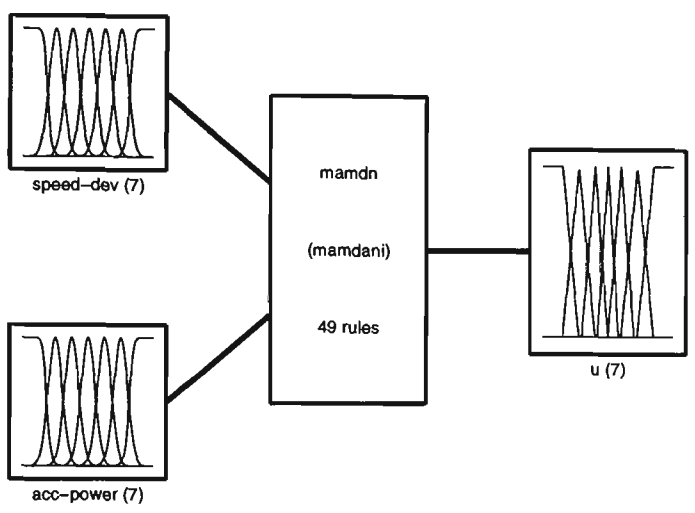


Fig. H.6. Fuzzy logic system mamdn.fis

```
function u_cont = FPSS(ip)
% Mamdani rule-based fuzzy PSS with fixed parameters to be used with SIMULINK as shown in Fig. H.5.

global vsmin vsmax spCOEF accCOEF uCOEF cont_FUZmat

deltaP = ip(1,1); SPdev = ip(2,1);

dPmax = 1; dPmin = -1;          % Maximum and minimum of normalized accelerating power in per unit
SPdmax = 1; SPdmin = -1;        % Maximum and minimun of normalized speed deviation in rad/sec.
SPdevN = spCOEF*SPdev;

if SPdevN > SPdmax
    SPdevN = SPdmax;
```

```
elseif SPdevN < SPdmin
    SPdevN = SPdmin;
end

deltaPN =  accCOEF*deltaP;
if deltaPN > dPmax
    deltaPN = dPmax;
elseif deltaPN < dPmin
    deltaPN = dPmin;
end

u_cont = uCOEF*evalfis([SPdevN,deltaPN], cont_FUZmat);
if u_cont<vsmin
    u_cont=vsmin;
elseif u_cont>vsmax
    u_cont=vsmax;
end
```

```
function u = HF_PSS()
% Hiyama-Based fuzzy PSS with fixed parameters to be used with SIMULINK in Fig. H.5. instead of FPSS

global  w0 x Kd Pm Pe spCOEF accCOEF
global vsmin vsmax deltaP u

SPdev = w0*x(1); SPdevN = spCOEF*SPdev;
deltaP = Pm - Pe - Kd*SPdev; deltaPN =  accCOEF*deltaP;
ip = [deltaPN; SPdevN];
u = Hiyamcog1(ip);

if u<vsmin
    u=vsmin;
elseif u>vsmax
    u=vsmax;
end
```

```
% Hiyamcog1.m
function [u] = Hiyamcog1(ip);

As = ip(1,1); SPdev = ip(2,1);

umax = 0.5;
Dr = 4;

% defining theta
if (SPdev == 0) & (As == 0)
    theta = 0;
elseif (SPdev == 0) & (As > 0)
    theta = 90;
elseif (SPdev < 0) & (As == 0)
    theta = 180;
```

```
elseif (SPdev == 0) & (As < 0)
    theta = 270;
else
    theta = (atan(As/SPdev))*180/pi;
end

if (SPdev < 0) & (As > 0)
    theta = theta + 180;
elseif (SPdev < 0) & (As < 0)
    theta = theta + 180;
elseif (SPdev > 0) & (As < 0)
    theta = 360 + theta;
end

R = sqrt((As^2) + (SPdev^2));
if R >= Dr
    Gc = 1;
else
    Gc = (1/Dr) * R;
end

% define p(theta)

theta0 =  90; Alpha = 90;
if (theta >= 0) & (theta < theta0)
    p = 0;
elseif (theta >= theta0) & (theta < (theta0 + Alpha/2))
    p = 2*((theta-theta0)/Alpha)^2;
elseif (theta >= (theta0+Alpha/2)) & (theta  < (theta0+Alpha))
    p = 1 - 2* ((theta-(theta0+Alpha))/Alpha)^2;
elseif (theta >= (theta0+Alpha)) & (theta < (theta0+180))
    p = 1;
elseif (theta >= (theta0+180)) & (theta < (theta0+180+Alpha/2))
    p = 1 - 2* ((theta-(theta0+180))/Alpha)^2;
elseif (theta >= (theta0+180+Alpha/2)) & (theta < (theta0+180+Alpha))
    p = 2* ((theta-(theta0+180+Alpha))/Alpha)^2;
elseif (theta >= (theta0+180+Alpha)) & (theta <= 360)
    p = 0;
end

% calculate control signal
u = Gc * (1 - 2*p) * umax;
```

Part II: Programs related to transient stability

(a) CPSS, FPSS, NFPSS and TFPSS

% wcfn_fault.m %

```
% Response of the Cogenerator after occurence of a fault
%(four responses: system without PSS; system with conventional PSS;
%system with rule-based fuzzy PSS; system with neuro-fuzzy PSS)

GLOB_init; simu_para; COG_data;

W_fault
deltaw = deltaw*180/pi;

C_fault
deltac = deltac*180/pi;

spCOEF = 0.2; accCOEF = 1.1;  % best for Pt0=0.9, Fp=0.9, Xe=0.15
uCOEF = 0.5;
F_fault
deltaf = deltaf*180/pi;

FN_fault
deltafn = deltafn*180/pi;

LW = 1.6;
MAINfigs4   % Figures for four responses: NO PSS, CPSS, FPSS, NFPSS
-------------------------------------------------------------------

% GLOB_init.m
% Calculating initial values of the Cogenerator plant
% This program should be run before the main programs

clear all; clear global
global  w0 x H Pm Kd Ra Zt Eb
global Rfd Xfd Xad R1d X1d R1q X1q R2q X2q
global Xd Xq Xdp Xqp Xddp Xqdp Xaqp Xaqsp Xadp Xadsp Xaqdp Xaqsdp Xaddp Xadsdp
global Et Et0 Eta id iq ed eq SYad SYaq
global Pe Q efd Efd0 DELTAt
global fuzMAT accCOEF spCOEF uCOEF deltaP Pt0 Qt0 Xta
global v1 v2 v3 v4 TR Vref KA Efd TA Kstab T1z T2z T1p T2p TW vsmin vsmax u
global v1_NNC Efd_NNC PNC a b c       % for AVR_NNC and tbpxNNC functions
global Pt0 Qt0 R1 R2 X1 X2
global cont_FUZmat spCOEF accCOEF uCOEF
global count CNT

% Parameters for solving differential equations
a(1)=0.5; b(1)=2; c(1)=a(1); a(2)=1-sqrt(0.5); b(2)=1; c(2)=a(2);
a(3)=1+sqrt(0.5); b(3)=1; c(3)=a(3); a(4)=1/6; b(4)=2; c(4)=0.5;
```

```
% simu_para.m
% Simulation parameters

global SpCoef dPCoef dPmCoef
global Nfinal tf dt dtf
dt = 0.005; Tfinal = 5; % time step and final time instant
Dt = 0.02;
tf = 0.2; % disturbance occuring instant
dtf=0.005; % time step during disturbance
tc=0.08, % duration of fault
tcD=0.2, % duration of disturbance

Nfinal = fix(tf/Dt)+fix(tc/dtf)+fix((Tfinal-tf-tc)/dt)+6;
NfinalD = Tfinal/dt;          % for programs based on disturbance

SpCoef = 0.95; dPCoef = 0.4;  % Coefficients for the predictor
dPmCoef = 0.95;   % For ANNPSS and adaptive fuzzy PSS's
```

```
% COG_data.m
global self_C
% Calculating initial values of the Cogenerator plant
% This program should be executed before the main programs

% fault type
self_C = 1; disp('self_C = 1: self-clearing fault')
%self_C = 0; disp('self_C = 0: switching off one line after fault occurrence')

% Transformer parameters
Rtr = 0.005; Xtr = 0.05;

% Network impedances
R1 = 0.06;  % Cases 1 and 2
R2 = 0.06;  % Cases 1 and 2
%R1 = 0.1;  % Case 3
%R2 = 0.1;  % Case 3

%X1 = 0.3; % Cases 1 and 2
%X1 = 0.9; % Case 3
X1 = 0.7;
 X2 = X1;

Eb0 = 1.0;  % Infinite bus voltage

%Pt0 = 0.9   % Case 1
Pt0 = 0.85   % Case 2
%Pt0 = 0.7   % Case 3

%Fp = 0.9   % Cases 1 and 2
%Fp = 0.7   % Case 3
```

```
Fp = 0.85

LAG = 1   % Case 1 and 3 (lagging power factor)
%LAG = -1  % Case 2 (-1 for leading power factor)
PHAY=sign(LAG)*acos(Fp);
Qt0 = Pt0*sin(PHAY)/Fp
%PF = LAG*Fp;

Ztr = Rtr + j*Xtr;
Z1 = R1 + j*X1; Z2 = R2 + j*X2;
Zeq = Z1*Z2/(Z1+Z2);
Xe0 = imag(Zeq)
Zt = Ztr+Zeq;
Re = real(Zt); Xta = imag(Zt); Zta = abs(Zt);

% Exciter, AVR and PSS parameters
KA=100;          % For Latrobe Cogenerator
%KA = 20;        % For Crown Cogenerator
vsmax=0.2; vsmin=-0.2;

Latrobe = 1; Crown = 0;
if Latrobe == 1
    Xd=2.64; Xdp=0.28; Xddp=0.2; Xq=1.32; Xqp=0.29; Xqdp=0.21; Xl=0.1;
    Ra=0.003; Td0p=5.79; Td0dp=0.044; Tq0p=1.2; Tq0dp=0.084; H = 3.8; Kd = 0.01;
elseif Crown == 1
    % Crown cogenerator standard parameters
    Xd=2.1877; Xdp=0.1763; Xddp=0.1384; Xq=1.0152; Xqp=0.18; Xqdp=0.1664; Xl=0.1;
    Ra=0.003; Td0p=4.14743; Td0dp=0.01483; Tq0p=0.6; Tq0dp=0.01387; H = 5; Kd = 0.02;
else
    Xd=2.2; Xdp=0.22; Xddp=0.2; Xq=1.01; Xqp=0.24; Xqdp=0.21; Xl=0.05;
    Ra=0.002; Td0p=4.5; Td0dp=0.05; Tq0p=1.3; Tq0dp=0.04; H = 5; Kd = 0.01;
end

f=50; w0=2*pi*f;

Ksd = 0.835; Ksq = 0.835;      % saturation coefficients
% Calculate fundamental parameters

Xad = Xd - Xl; Xaq = Xq - Xl;
Xads = Ksd*Xad; Xaqs = Ksq*Xaq;
Xds = Xads + Xl; Xqs = Xaqs + Xl;
Xfd = Xad*(Xdp-Xl)/(Xad+Xl-Xdp);
X1q = Xaq*(Xqp-Xl)/(Xaq+Xl-Xqp);
X1d = Xad*Xfd*(Xl-Xddp)/((Xad+Xfd)*(Xddp-Xl)-Xad*Xfd);
X2q = Xaq*X1q*(Xl-Xqdp)/((Xaq+X1q)*(Xqdp-Xl)-Xaq*X1q);
Rfd = (Xad+Xfd)/(Td0p*w0);
R1d = (X1d+Xad*Xfd/(Xad+Xfd))/(Td0dp*w0);
R1q = (Xaq+X1q)/(Tq0p*w0);
R2q = (X2q+Xaq*X1q/(Xaq+X1q))/(Tq0dp*w0);

% Calculate Et and It0
```

```
It0 = 1;              % Estimation of the current through network to be 1.0 pu
for k = 1:10
    Poloss = (It0^2)*Re; Ir = Pt0 - Poloss;
    tanPHAY = tan(PHAY); A = Re*tanPHAY - Xta; B = Eb0; C = tanPHAY*(Ir*Eb0+Re*Ir^2)-Xta*Ir^2;
    Ix = (-B+sqrt(B^2-4*A*C))/(2*A);
    Et = (Eb0-Xta*Ix+Re*Ir) + j*(Xta*Ir+Re*Ix);
    Et0 = abs(Et); %deltat=angle(Et);
    EtR = real(Et); EtX = imag(Et);
    deltat = atan2(EtX,EtR);
    It0 = sqrt(Ir^2+Ix^2);
end

% Calculating operating point values

deltai=atan((Xqs*It0*cos(PHAY)-Ra*It0*sin(PHAY))/(Et0+Ra*It0*cos(PHAY)+Xqs*It0*sin(PHAY)));
ed0=Et0*sin(deltai); eq0=Et0*cos(deltai);
id0=It0*sin(deltai+PHAY); iq0=It0*cos(deltai+PHAY);
ifd0=(eq0+Ra*iq0+Xds*id0)/Xads;
efd0=Rfd*ifd0;
Ifd0=Xad*ifd0;
Efd0=(Xad/Rfd)*efd0;
delta0=deltai+deltat;

SYfd0=(Xads+Xfd)*ifd0-Xads*id0;
SY1d0=Xads*(ifd0-id0); SY1q0=-Xaqs*iq0; SY2q0=SY1q0;
Xadsp=1/(1/Xads+1/Xfd); Xaqsp=1/(1/Xaqs+1/X1q);
Xadsdp=1/(1/Xads+1/Xfd+1/X1d); Xaqsdp=1/(1/Xaqs+1/X1q+1/X2q);
SYad0=Xadsdp*(-id0+SYfd0/Xfd+SY1d0/X1d); SYaq0=Xaqsdp*(-iq0+SY1q0/X1q+SY2q0/X2q);
i1d0=0; i1q0=0; i2q0=0; dwr0 = 0;
Pe0=Pt0+Ra*It0^2; Pm0 = Pe0;
 Vref0=Efd0/KA+Et0;
```

% W_fault.m %

```
% Response of the cogenerator without any PSS after occurence of a fault

global Efd0 v10 Eb Vref u_cont Eta Pe Q DELTAt tol x0 Pm Pe H
clear tout; clear yout;
Pedot = 0;

    main1
    [tout,yout,J,k] = mloopw_ST(k0,yout0,J0,t0,Tf,Efd0,Vref,DELTAt);
    main2
    [tout,yout,J,k] = mloopw_ST(k0,yout0,J0,t0,Tf,Efd0,Vref,DELTAt);
    yout = [youtp; yout(k0:length(yout),:)];
    tout = [toutp, tout(k0:length(tout))];
    dwrw = yout(:,1); deltaw = yout(:,2); Vtw = yout(:,7); x2sw = (Pm - yout(:,9))/(2*H);
    Qw = yout(:,10); usw = yout(:,11); PEw = yout(:,9);
    Jw = J
```

% main1.m %

```
% First part of the main programs for simulating the power system after occurrence of a fault

clear tout; clear yout;
t0=0; tout(1)=t0; x0 = [dwr0; delta0; SYfd0; SY1d0; SY1q0; SY2q0];
Pe=Pt0+Ra*It0^2; Pm = Pe; Q=Qt0;
Efd = Efd0;
SYad=SYad0; SYaq=SYaq0;
Eta=Et0; v10 = Eta; Etas(1) = Et0;
u_cont = 0;
yout0 = [w0*x0(1); x0(2);x0(3);x0(4);x0(5);x0(6); Eta; Efd0; Pe; Q; u_cont];

% Steady state
disp('Steady state')
% Initialization
    Vref=Efd0/KA+v10; Eb = Eb0; t = t0;
    k0 = 1;
    J0 = 0;
    DELTAt = 0.02; tol = 0.01; Tf = tf;
    k = k0;
    yout(k,:) = yout0.';
    while t <= Tf
        k = k+1;
        tout(k) = t;
        t = t + DELTAt;
        yout(k,:) = yout0.';
    end

    youtp = yout; toutp = tout;
    t0 = tout(k);
    yout0 = (yout(k,:)).';
    k0 = k;

% Subtransient period
disp('Sub-transient period')
    DELTAt = 0.005;
    Zt=Ztr; Xta = imag(Zt); Eb = 0;
    Tf = tf+tc;
```

% main2.m %

```
% Second part of the main programs for simulating the power system after occurrence of a fault
    yout = [youtp; yout(k0:length(yout),:)];
    youtp = yout;
    tout = [toutp, tout(k0:length(tout))];
    toutp = tout;
    t0 = tout(k)
    yout0 = (yout(k,:)).';
    k0 = k;
    J0 = J;

% Transient period
```

```
disp('Transient period')
    % If faulty line is switched off after occurrence of the fault
    Zt=Ztr+Z1; Xta = imag(Zt);

    % If the faulty line is re-energised after clearing the fault
    %Zeq = Z1*Z2/(Z1+Z2); Xe = imag(Zeq); Zt = Ztr+Zeq; Xta = imag(Zt);

    Eb = Eb0; DELTAt = 0.005;
    Tf = Tfinal-DELTAt;
```

% mloopw_ST.m %

```
% The main loop consisting of the AVR & exciter and the synchronous machine

function [tout,yout,J,k] = mloopw_ST(k0,yout0,J0,t0,Tf,Efd0,Vref,DELTAt)

global w0 u_cont Eta Pe Q x0 Efd tf v10
t = t0; J = J0; k = k0; x = x0;
tout(k) = t; yout(k,:) = yout0.';

while t <= Tf
    [t,v1,Efd,Flag] = trns_avr(t0,Efd0,Vref,u_cont);
        if Flag == 1
            v10 = v1; Efd0 = Efd; clear t; delPp = Pe;
            [t,x,Flag] = onest_de('sync_mST', t0, x0);
            if Flag == 1
                delP = Pe; Pedot = (delP-delPp)/DELTAt;
                x0 = x; t0 = t;
                YY = [w0*x(1); x(2); x(3); x(4); x(5); x(6); Eta; Efd; Pe; Q; u_cont];
                k = k+1;
                tout(k) = t; yout(k,:) = YY.';
                indf = abs(w0*x(1))*(t-tf);
                J = J + indf;
            end
        end
end
```

```
% trns_avr.m
function [t,v1,Efd,Flag] = trns_avr(t0,Efd0,Vref,u_cont)
global Rfd Xad VE DELTAt KA v10
EFmax = 5; EFmin = -2;        % For Latrobe Cogenerator

[t,v1,Flag] = onest_de('trsducer', t0, v10);
if Flag == 1
    VE = Vref-v1+u_cont;
    beneath = 1;
    if beneath == 1
        [t,Efd,Flag] = onest_de('avr_sat', t0, Efd0);
        if Flag == 1
            if Efd<EFmin
                Efd=EFmin;
```

```
            elseif Efd>EFmax
                Efd=EFmax;
            end
        end
    else
        Efd = KA*VE;
    end
end
```

```
% onest_de.m
function [t,y,Flag] = onest_de(ypfun, t0, y0)

% onest_de solves differential equations for only one sampling time. It integrates a system of ordinary differential
% equations using 4th and 5th order Runge-Kutta formulas.
% By N. Hosseinzadeh, 2/12/96. Revised from ODE45 in MATLAB.

global DELTAt tol

% The Fehlberg coefficients:
alpha = [1/4  3/8  12/13  1  1/2]';
beta  = [ [   1     0     0     0     0    0]/4
        [   3     9     0     0     0    0]/32
        [ 1932  -7200   7296    0     0    0]/2197
        [ 8341 -32832  29440  -845    0    0]/4104
        [-6080  41040 -28352  9295  -5643   0]/20520 ]';
gamma = [ [902880  0  3953664  3855735  -1371249  277020]/7618050
        [ -2090  0    22528    21970    -15048  -27360]/752400 ]';
pow = 1/5;

% Initialization
t = t0; y = y0(:);
f = zeros(length(y),6);

  % Compute the slopes
  temp = feval(ypfun,t,y);
  f(:,1) = temp(:);
  for j = 1:5
    temp = feval(ypfun, t+alpha(j)*DELTAt, y+DELTAt*f*beta(:,j));
    f(:,j+1) = temp(:);
  end

  % Estimate the error and the acceptable error
  delta = norm(DELTAt*f*gamma(:,2),'inf');
  tau = tol*max(norm(y,'inf'),1.0);

  % Decrease the step size if the error is not acceptable
  if delta > tau
  disp('Singularity likely.')
    DELTAt = 0.8*DELTAt*(tau/delta)^pow;
    Flag = 0;
  else
```

```
 % Update the solution only if the error is acceptable
   t = t + DELTAt;
   y = y + DELTAt*f*gamma(:,1);
   Flag = 1;
 end
```

```
% sync_mST.m
% differential equations of a synchronous machine dynamics after occurrence of a fault

function [xp] = sync_mST(t,x)
global Rfd Xfd R1d X1d R1q X1q R2q X2q Xad Xddp Xqdp Xaqdp Xaqsdp Xaddp Xadsdp
global H Pm Kd Ra Zt w0 Eb Efd DELTAt Pe Q Eta id iq ed eq SYad SYaq self_C

if self_C == 1
    Vt_low = 0.7;
    Vt_high = 0.9;
else
    Vt_low = 0.9;
    Vt_high = 1.1;
end

md = (1/(Vt_low - Vt_high))*(Xaddp-Xadsdp);
mq = (1/(Vt_low - Vt_high))*(Xaqdp-Xaqsdp);

if (Eta > Vt_low & Eta < Vt_high)
    XadDP = md*(Eta-Vt_low)+Xaddp;
    XaqDP = mq*(Eta-Vt_low)+Xaqdp;
elseif (Eta>=Vt_high)
    XaqDP = Xaqsdp;
    XadDP = Xadsdp;
else
    XaqDP = Xaqdp;
    XadDP = Xaddp;
end

efd = Rfd*Efd/Xad;
Eddp = -XaqDP*(x(5)/X1q+x(6)/X2q);
Eqdp = XadDP*(x(3)/Xfd+x(4)/X1d);
ERdp = Eddp*sin(x(2))+Eqdp*cos(x(2));
EIdp = Eqdp*sin(x(2))-Eddp*cos(x(2));
R_RR = (Xddp - Xqdp)*sin(x(2))*cos(x(2)) + Ra;
R_II = (Xqdp - Xddp)*sin(x(2))*cos(x(2)) + Ra;
X_RI = Xddp*(cos(x(2))^2) + Xqdp*(sin(x(2))^2);
X_IR = Xddp*(sin(x(2))^2) + Xqdp*(cos(x(2))^2);
mat_A = [-R_RR X_RI; -X_IR -R_II];
Edp = ERdp+j*EIdp; Zdp = Ra+j*Xddp;
for k = 1:3
    It_ph = (Edp-Eb)/(Zdp+Zt);
    IR = real(It_ph); II = imag(It_ph);
    ER_EI = mat_A*[IR;II] + [ERdp;EIdp];
```

```
    ER = ER_EI(1); EI = ER_EI(2);
    Et = ER + j*EI;
    Zdp = (Edp-Et)/It_ph;
end

ed = ER*sin(x(2))-EI*cos(x(2));
eq = EI*sin(x(2))+ER*cos(x(2));
DEN = Ra^2 + Xddp*Xqdp;
id = (Ra*(Eddp-ed)+Xqdp*(Eqdp-eq))/DEN;
iq = (Ra*(Eqdp-eq)-Xddp*(Eddp-ed))/DEN;
SYad = XadDP*(-id+x(3)/Xfd+x(4)/X1d);
SYaq = XaqDP*(-iq+x(5)/X1q+x(6)/X2q);
Pe = SYad*iq-SYaq*id;
Q = eq*id - ed*iq;
Eta = abs(Et);

xp(1) = (Pm-Pe-Kd*x(1))/(2*H);
xp(2) = w0*x(1);
xp(3) = w0*(efd+(SYad-x(3))*Rfd/Xfd);
xp(4) = w0*(SYad-x(4))*R1d/X1d;
xp(5) = w0*(SYaq-x(5))*R1q/X1q;
xp(6) = w0*(SYaq-x(6))*R2q/X2q;
```

```
% transducer.m
function v1p = trsducer(t,v1)
global Eta
TR=0.015;
v1p = (Eta-v1)/TR;
```

```
% AVR_sat.m
function Efdp = avr_sat(t,Efd)
global KA VE

% AVR time constant
TA=0.05;
Efdp = (KA*VE-Efd)/TA;
```

% C_fault.m %

```
% Response of the Cogenerator after occurence of a fault (system with conventional PSS);

global Efd0 Efd v10 v0 u_cont Eta Et0 Pe Q DELTAt Pe_ba x0 Pm
v0 = [0 0 0];
main1
[tout,yout,J,k] = Cloop_ST(k0,yout0,J0,t0,Tf,Efd0,Vref,u_cont,DELTAt);
main2
[tout,yout,J,k] = Cloop_ST(k0,yout0,J0,t0,Tf,Efd0,Vref,u_cont,DELTAt);
yout = [youtp; yout(k0:length(yout),:)];
```

```
tout = [toutp, tout(k0:length(tout))];
dwrc = yout(:,1); deltac = yout(:,2); Vtc = yout(:,7); PEc = yout(:,9); Qc = yout(:,10); usc = yout(:,11);
Jc = J
```

% Cloop_ST.m %

```
% The main loop for conventional controller
% consisting of the CPSS & the AVR & exciter and the synchronous machine

function [tout,yout,J,k] = Cloop_ST(k0,yout0,J0,t0,Tf,Efd0,Vref,u_cont,DELTAt)
global w0 Eta Pm Pe Q x0 x v10 v0 Efd tf vsmin vsmax count CNT

CNT = 30; count = 1;

t = t0; J = J0; x = x0; k = k0;
tout(k) = t; yout(k,:) = yout0.';

while t <= Tf
    % Conventional PSS
    [t,v] = onest_de('CPSS', t0, v0);
    u_cont = v(3);
    if u_cont<vsmin
        u_cont=vsmin;
    elseif u_cont>vsmax
        u_cont=vsmax;
    end

    YY = [w0*x(1); x(2); x(3); x(4); x(5); x(6); Eta; Efd; Pe; Q; u_cont];
    [t,v1,Efd,Flag] = trns_avr(t0,Efd0,Vref,u_cont);
    if Flag == 1
        v10 = v1; Efd0 = Efd;
        [t,x,Flag] = onest_de('sync_mST', t0, x0);
        if Flag == 1
            x0 = x; t0 = t;
            k = k+1;
            tout(k) = t; yout(k,:) = YY.'; uout(k) = u_cont; Vout(k) = Eta;
            indf = abs(w0*x(1))*(t-tf);
            J = J + indf;
        end
    end
end
```

```
function vp = CPSS(t0,v0)
% function that solves the state space equations of the conventional PSS

global Pm Pe x0 Kd H

 % The parameters of the CPSS
Kstab = 20;TW = 10;
T1z = 0.12; T1p = 0.014;
```

```
T2z = 0.12; T2p = 0.014;

v = v0;
vp(1) = Kstab*(Pm-Pe-Kd*x0(1))/(2*H)-v(1)/TW;
vp(2) = (T1z*(Kstab*(Pm-Pe-Kd*x0(1))/(2*H)-v(1)/TW)+v(1)-v(2))/T1p;
vp(3) = (1/T2p)*(T2z*(T1z*(Kstab*(Pm-Pe-Kd*x0(1))/(2*H)-v(1)/TW)+v(1)-v(2))/T1p+v(2)-v(3));
```

% F_fault.m %

```
% Response of the Cogenerator after occurence of a fault (system with a fixed-parameters fuzzy PSS);

global Efd0 Efd v10 u_cont Eta Et0 Pe Q DELTAt Pe_ba x0 Pm Eb Zt cont_FUZmat spCOEF accCOEF uCOEF

% calling Mamdani rule-based fuzzy controller
cont_FUZmat = readfis('matlab/final/mamdn');

main1
[tout,yout,J,k] = FUZloop_ST(k0,yout0,J0,t0,Tf,Efd0,Vref,u_cont,DELTAt);
main2
[tout,yout,J,k] = FUZloop_ST(k0,yout0,J0,t0,Tf,Efd0,Vref,u_cont,DELTAt);
yout = [youtp; yout(k0:length(yout),:)];
tout = [toutp, tout(k0:length(tout))];

dwrf = yout(:,1); deltaf = yout(:,2); Vtf = yout(:,7); Efdf = yout(:,8); PEf = yout(:,9); Qf = yout(:,10); usf = yout(:,11);
Jf = J
```

% FUZloop_ST.m %

```
% The main loop for fuzzy controller
% consisting of the FPSS & AVR & exciter and the synchronous machine

function [tout,yout,J,k] = FUZloop_ST(k0,yout0,J0,t0,Tf,Efd0,Vref,u_cont,DELTAt)
global w0 Eta Pm Pe Q x0 x Efd v10 tf vsmin vsmax count CNT

CNT = 30; count = 1;
t = t0; J = J0; x = x0; k = k0;
tout(k) = t; yout(k,:) = yout0.';

while t <= Tf
    u_cont = FPSS;                  % Fixed-parameters fuzzy PSS
    YY = [w0*x(1); x(2); x(3); x(4); x(5); x(6); Eta; Efd; Pe; Q; u_cont];
    [t,v1,Efd,Flag] = trns_avr(t0,Efd0,Vref,u_cont);
    if Flag == 1
        v10 = v1;
        Efd0 = Efd;
        [t,x,Flag] = onest_de('sync_mST', t0, x0);
        if Flag == 1
            x0 = x; t0 = t;
            k = k+1;
            tout(k) = t; yout(k,:) = YY.'; uout(k) = u_cont; Vout(k) = Eta;
            indf = abs(w0*x(1))*(t-tf);
```

```
            J = J + indf;
        end
    end
end
```

% FN_fault.m %

```
% Response of the Cogenerator after occurence of a fault (system with a neuro-fuzzy PSS);

global Efd0 Efd v10 u_cont Eta Et0 Pe Q DELTAt Pe_ba x0 Pm Eb Zt cont_FUZmat

% calling Mamdani rule-based fuzzy controller
cont_FUZmat = readfis('matlab/FLPSS/wcff_PSS/mamdn');

% Calculating accCOEF and spCOEF with a neural network
[spCOEF,accCOEF] = KdKpCompute(Pt0,Qt0,Xe0)

main1
[tout,yout,J,k] = FUZloop_ST(k0,yout0,J0,t0,Tf,Efd0,Vref,u_cont,DELTAt);
main2
[tout,yout,J,k] = FUZloop_ST(k0,yout0,J0,t0,Tf,Efd0,Vref,u_cont,DELTAt);

yout = [youtp; yout(k0:length(yout),:)];
tout = [toutp, tout(k0:length(tout))];

dwrfn = yout(:,1); deltafn = yout(:,2); Vtfn = yout(:,7); PEfn = yout(:,9); Qfn = yout(:,10); usfn = yout(:,11);
Jfn = J
```

```
% KdKpCompute.m
% (Computing Kd and Kp with a neural network)

function [spCOEF,accCOEF] = KdKpCompute(Pg,Qg,Xe);
P=[Pg;Qg;Xe];
load matlab/final/wcfn_pss/TEMPfiles/W1_KdKp
load matlab/final/wcfn_pss/TEMPfiles/b1_KdKp
load matlab/final/wcfn_pss/TEMPfiles/W2_KdKp
load matlab/final/wcfn_pss/TEMPfiles/b2_KdKp

spCOEF_min = 0.03; spCOEF_max = 1.5;
accCOEF_min = 0.1; accCOEF_max = 5;

KdKp = simuff(P,W1,b1,'tansig',W2,b2,'purelin');
spCOEF = KdKp(1,1); accCOEF = KdKp(2,1);
if spCOEF < spCOEF_min
    spCOEF = spCOEF_min;
elseif spCOEF > spCOEF_max
    spCOEF = spCOEF_max;
end

if accCOEF < accCOEF_min
```

```
    accCOEF = accCOEF_min;
elseif accCOEF > accCOEF_max
    accCOEF = accCOEF_max;
end
```

```
% KpKdtrain.m
% (training a neural network to compute Kd and Kp)
% This program should be executed before running KdKpCompute.m

load matlab/final/wcfn_pss/TEMPfiles/PtQtXe_LAT.mat
load matlab/final/wcfn_pss/TEMPfiles/KpKd_LAT.mat
P = PtQtXe; T = KpKd;

% Random initialization
%S1 = 12;
%[W1,b1,W2,b2] = initff(P,S1,'tansig',T,'purelin');
%save matlab/final/wcfn_pss/TEMPfiles/W1init_LAT W1
%save matlab/final/wcfn_pss/TEMPfiles/b1init_LAT b1
%save matlab/final/wcfn_pss/TEMPfiles/W2init_LAT W2
%save matlab/final/wcfn_pss/TEMPfiles/b2init_LAT b2

% Initializing with a specific set of weights
load matlab/final/wcfn_pss/TEMPfiles/W1init_LAT
load matlab/final/wcfn_pss/TEMPfiles/b1init_LAT
load matlab/final/wcfn_pss/TEMPfiles/W2init_LAT
load matlab/final/wcfn_pss/TEMPfiles/b2init_LAT

Lev_Mar = 1        Either 0 (backpropagation training) or 1 (Levenberg-Marquardt training)
if Lev_Mar == 1
    disp_freq = 20; max_epoch = 80; err_goal = 0.1; grad_min = 1e-8;
    mu = 0.01; mu_inc = 10; mu_dec = 0.9; mu_max = 10;
    figure(1);
    tp = [disp_freq max_epoch err_goal grad_min mu mu_inc mu_dec mu_max];
    [W1,b1,W2,b2,te,tr] = trainlm(W1,b1,'tansig',W2,b2,'purelin',P,T,tp);
else
    disp_freq = 200; max_epoch = 5000; err_goal = 0.05; lr = 0.001;
    momentum = 0.95; err_ratio = 1.1; lr_inc = 1.5; lr_dec = 0.95;
    figure(1); clf;
    tp = [disp_freq max_epoch err_goal lr];
    [W1,b1,W2,b2,epochs,tr] = trainbp(W1,b1,'tansig',W2,b2,'purelin',P,T,tp);
end

save matlab/final/wcfn_pss/TEMPfiles/W1_KdKp W1
save matlab/final/wcfn_pss/TEMPfiles/b1_KdKp b1
save matlab/final/wcfn_pss/TEMPfiles/W2_KdKp W2
save matlab/final/wcfn_pss/TEMPfiles/b2_KdKp b2

P = [0.8 ; 0.4 ; 0.2]
KdKp = simuff(P,W1,b1,'tansig',W2,b2,'purelin')
```

```
% FF_fault.m
% Response of the Cogenerator after occurence of a fault (system with a tuned-fuzzy PSS);

global Pe_ba x0 Pm Eb Zt cont_FUZmat uCOEF

% calling Mamdani rule-based fuzzy controller and fuzzy tuner
cont_FUZmat = readfis('matlab/final/mamdn');
tune_FUZmat = readfis('matlab/final/wcff_pss/scaling_Lat');

output = evalfis([Pt0,Qt0,Xe0], tune_FUZmat);
spCOEF = output(1), accCOEF = output(2)

main1
[tout,yout,J,k] = FUZloop_ST(k0,yout0,J0,t0,Tf,Efd0,Vref,u_cont,DELTAt);
main2
[tout,yout,J,k] = FUZloop_ST(k0,yout0,J0,t0,Tf,Efd0,Vref,u_cont,DELTAt);

yout = [youtp; yout(k0:length(yout),:)];
tout = [toutp, tout(k0:length(tout))];
usaf = [usafp, usaf(k0:length(usaf))];
Etas = [Etasp, Etas(k0:length(Etas))];

dwrfn = yout(:,1); deltafn = yout(:,2); Vtfn = yout(:,7); PEfn = yout(:,9); Qfn = yout(:,10); usfn = yout(:,11);
Jff = J
```

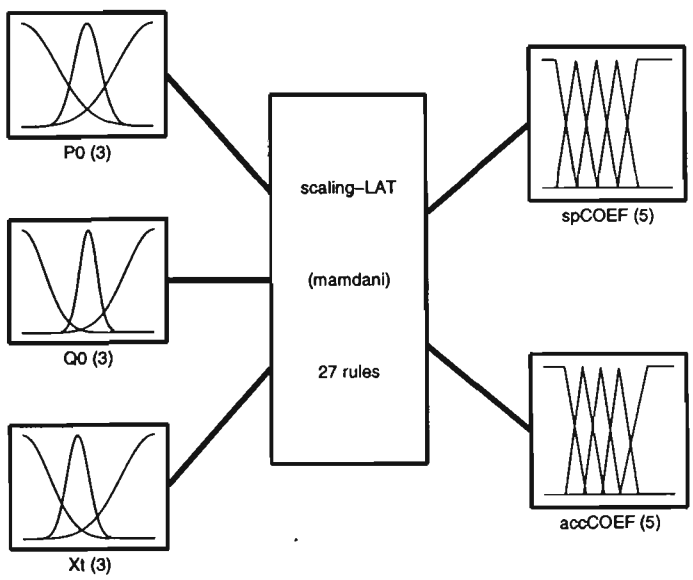


Fig. H.7. Fuzzy logic system scaling_LAT.fis

(b) ANNPSS

% wcAdNN_fault_Efd2.m %

```
% Three responses: system without PSS, system with CPSS and system with AdNN PSS
% Adaptive NN PSS with input dw and its delays, dP and its delays
%% NNM is trained every PNM samples and NNC is trained every PNC samples %%

GLOB_init; simu_para; COG_data

W_fault
deltaw = deltaw * 180/pi;  % converting radians to degrees

C_fault
deltac = deltac * 180/pi;  % converting radians to degrees

NNCinit
AdNN_fault_Efd2
deltaN = deltaN*180/pi;

AdNNfigs3

save matlab/final/AdNN_pss/Tempf/TEMPm_Lat WM1 BM1 WM2 BM2;
```

```
% NNCinit.m
% (Neural Network Controller initialization for using before AdNN_fault_Efd2.m)

% Initializing the NNC
PC = [-2 2;-2 2;-1 1;-1 1];
SC1 = 5;
MAX_init = 0.001;
[WC1,BC1,WC2,BC2] = initff(PC,SC1,'tansig',1,'purelin');
WC1 = max(-MAX_init,min(MAX_init,WC1));
WC2 = max(-MAX_init,min(MAX_init,WC2));
BC1 = zeros(SC1,1);
BC2  = zeros(1,1);

save matlab/final/AdNN_pss/Tempf/tempc0 WC1 BC1 WC2 BC2;
```

% AdNN_fault_Efd2.m (AdNN PSS for cogenerator) %

```
% Adaptive NN PSS with input dw and its delays, dP and its delays
%% NNM is trained every PNM samples and NNC is trained every PNC samples %%

eSIGN = 1;

% Parameters for training NNM and NNC and pattern initialization
NNMC_Efd_PATTinit2
load matlab/final/AdNN_pss/Tempf/TEMPm_Lat;
t(1)=0;  x0 = [dwr0; delta0; SYfd0; SY1d0; SY1q0; SY2q0];
dwrN(1)=w0*x0(1); deltahs(1) =x0(2);deltaN(1)=x0(2); deltah=x0(2); EFDN(1)=Efd0; usN(1) = 0;
```

```
SYfd(1)=x0(3); SY1d(1)=x0(4); SY1q(1)=x0(5); SY2q(1)=x0(6);
Pe=Pe0; Pm = Pe0; PEN(1)=Pe; Q=Qt0; QN(1)=Qt0; VtN(1)=Et0; dEtNs(1)=0;
dPg(1) = Pm-Pe-Kd*x0(1);
efd=efd0;
SYad=SYad0; SYaq=SYaq0; Eta=Et0; Vref=Vref0;
v1=Et0; Efd=Efd0;
v1_NNC=Et0; Efd_NNC=Efd0;

JN=0;
ind_pow = 1;
for n = 1 : tf/Dt;
    PEN(n+1) = Pe; QN(n+1) = Q; EFDN(n+1) = Efd0; dwrN(n+1) = w0*x0(1); deltaN(n+1) = x0(2);
    deltahs(n+1) = x0(2); VtN(n+1) = Et0; dEtNs(n+1) = 0; dPg(n+1) = dPg(1);
    t(n+1)=t(n)+Dt;
end

x = x0;
Zt=Ztr; Xta = imag(Zt); Eb = 0; DELTAt = dtf;

for n = tf/Dt+1 : (tf/Dt+1)+(tc/dtf);
    deltaP = Pm - Pe; dPg(n+1) = deltaP;
    NNM_Efd_pattern2        % Patterns for NNM
    NNM_Efd2                % Neural Network Identifier
    NNC_Efd_pattern2        % Patterns for NNC
    NNC_Efd_f2              % Fixed Neural Network Controller
    usN(n+1)=u;
    AVR;
    COGplantST;
    Eta = abs(Et);
    PEN(n+1) = Pe; QN(n+1) = Q; EFDN(n+1) = Efd; dwrN(n+1) = w0*x(1);
    deltaN(n+1) = x(2);VtN(n+1) = Eta; dEtNs(n+1) = Eta-Et0; t(n+1)=t(n)+dtf;
    indf = abs(dwrN(n+1))*(t(n+1)-tf)^ind_pow;
    JN = JN + indf;
end

Zt=Ztr+Z1; Xta = imag(Zt);
%Zt = Ztr+(Z1*Z2/(Z1+Z2)); Xta = imag(Zt);
Eb = Eb0;
DELTAt = dt;

for n = (tf/Dt+1)+(tc/dtf)+1 : Nfinal-1;
    deltaP = Pm - Pe; dPg(n+1) = deltaP;
    NNM_Efd_pattern2        % Patterns for NNM
    NNM_Efd2                % Neural Network Identifier
    NNC_Efd_pattern2        % Patterns for NNC
    NNC_Efd2                % Adaptive Neural Network Controller
    usN(n+1)=u;
    AVR;
    COGplantST;
    Eta = abs(Et);
    PEN(n+1) = Pe; QN(n+1) = Q; EFDN(n+1) = Efd; dwrN(n+1) = w0*x(1);
```

```
    deltaN(n+1) = x(2); VtN(n+1) = Eta; dEtNs(n+1) = Eta-Et0; t(n+1)=t(n)+dt;
    indf = abs(dwrN(n+1))*(t(n+1)-tf)^ind_pow;
    JN = JN + indf;
end

JN = JN
```

% NNMC_Efd_PATTinit2.m %

```
% Parameters and pattern initialization for NNM and NNC

global G
PNM = 10; PNC = 3; ORD_pat = 2;
% Parameters for training NNM and NNC
tpm = [20 PNM*1e-8 .0005];      % [max. no. of epochs; error goal; learning rate]
lr = 0.005; lr_max = 10*lr;
tpc = [10 PNC*1e-8 lr];         % [max. no. of epochs; error goal; learning rate]

% Normalising coefficient for speed deviation in rad/sec
alpha = 0.5; beta  = 1; zeta = 1;

% Normalising coefficient for accelerating power
gamma0 = 5; gamma = gamma0; gamma_max = 2*gamma0;

% Controller's gain
G0 = 0.9; G = G0; Gmax = 3*G0;

% Patterns initialization

dEfdm = zeros(ORD_pat+1,1); dEfdmp =  zeros(ORD_pat+1,PNM);
dwm = zeros(ORD_pat,1); dwmp = zeros(ORD_pat,PNM);
detm = zeros(ORD_pat,1); detmp = zeros(ORD_pat,PNM);
dpm = zeros(ORD_pat,1); dpmp = zeros(ORD_pat,PNM);

TM = zeros(1,PNM);
dwcp = zeros(ORD_pat,PNC);
dpcp = zeros(ORD_pat,PNC);
detcp = zeros(ORD_pat,PNC);

dw0 = 0; dw1 = 0; dp0 = 0; dp1 = 0; det0 = 0; det1 = 0;

Dr = 0.4;
% if distance from origin in the phase plane (D) is less than Dr learning rate will increase accordingly

GdP_Coef = 1.0;  % coefficient of dP in calculating D
```

% NNM_Efd_pattern2.m %

```
% creating patterns for NNM

NNMpat = rem(n,PNM);
```

```
if NNMpat == 0
    NNMpat = PNM;
end
DELAY = 0;
for L = 1 : ORD_pat
    dwmp(L,NNMpat) = alpha*dwrN(n-L);
    detmp(L,NNMpat) = dEtNs(n-L);
    dpmp(L,NNMpat) = gamma*dPg(n-L);
    dwm(L) = alpha*dwrN(n-L+1);
    detm(L) = dEtNs(n-L+1);
    dpm(L) = gamma*dPg(n-L+1);
end

if n > 1+DELAY
    dEfdmp(1,NNMpat) = EFDN(n-1-DELAY)-Efd0;
else
    dEfdmp(1,NNMpat) = 0;
end

if n >= 1+DELAY
    dEfdm(1,1) = EFDN(n-DELAY)-Efd0;
else
    dEfdm(1,1) = 0;
end

for L = 1 : ORD_pat+1
    dEfdmp(L,NNMpat) = dEfdmp(1,NNMpat);
    dEfdm(L,1) = dEfdm(1,1);
end

TM(1,NNMpat) = dwrN(n);
```

% NNM_Efd2.m %

```
% Neural Network Modeller

if NNMpat == PNM
    PM = [dEfdmp;dwmp;detmp;dpmp];
    WM1,WM2] = mytbp2(WM1,BM1,'tansig',WM2,BM2,'purelin',PM,TM,tpm);
end
PMnext = [dEfdm;dwm;detm;dpm];
dwhs(n+1) = simuff(PMnext,WM1,BM1,'tansig',WM2,BM2,'purelin');
```

% NNC_Efd_pattern2.m %

```
% Creating patterns for NNC

global SpCoef dPCoef

NNCpat = rem(n,PNC);
if NNCpat == 0
```

```
    NNCpat = PNC;
end

for L = 1 : ORD_pat
    dEfdcp(L,NNCpat) = EFDN(n);
end

for L = 1 : ORD_pat
    dwcp(L,NNCpat) = alpha*dwrN(n-L+1);
    detcp(L,NNCpat) = dEtNs(n-L+1);
    dpcp(L,NNCpat) = gamma*dPg(n-L+1);
end

beneath = 0;
if beneath == 0
    dw_desired_f(1,NNCpat) = dwrN(n);
    dw_desired(1,NNCpat) = SpCoef*dwrN(n) - dPCoef*dPg(n);
else
    dw_desired_f(1,NNCpat) = dwrN(n);
    XXp = [dwrN(n); dPg(n)];
    [time,xm1] = ONEst_deO('MsDo', 0, XXp, DELTAt);
    dw_desired(1,NNCpat) = xm1;
end
```

% NNC_Efd_f2.m %

```
% Neural Network fixed controller

if NNCpat == PNC
    TMh = [dw_desired_f];
    dwps(n+2-PNC:n+1) = dw_desired_f/zeta;
    Pc1 = [dwcp(1:2,:);dpcp(1:2,:)];
    Pc2 = [dEfdcp;dwcp;detcp;dpcp];
    [WC1,WC2] = tbpxNNC_N2(WC1,BC1,'tansig',WC2,BC2,'purelin', ...
                WM1,BM1,'tansig',WM2,BM2,'purelin',Pc1,Pc2,TMh,eSIGN,n,DELAY,tpc);
end
if  n > tf/dt+1
    D = sqrt((w0*x(1))^2 + (GdP_Coef*deltaP)^2);
    if D < Dr
        G = G * (D/Dr);
    end
end

dw0 = alpha*dwrN(n);  dp0 = gamma*dPg(n);
dw1 = alpha*dwrN(n-1);  dp1 = gamma*dPg(n-1);
det0 = dEtNs(n); det1 = dEtNs(n-1);
P1 = [dw0;dw1;dp0;dp1];
u = simuff(P1,WC1,BC1,'tansig',WC2,BC2,'purelin');
u = min(vsmax,max(vsmin,G*u));
G = G0;
```

% NNC_Efd2.m %

```
% Neural Network Controller

if NNCpat == PNC
    TMh = [dw_desired];
    dwps(n+2-PNC:n+1) = dw_desired/zeta;
    Pc1 = [dwcp(1:2,:);dpcp(1:2,:)];
    Pc2 = [dEfdcp;dwcp;detcp;dpcp];
    [WC1,WC2] = tbpxNNC_N2(WC1,BC1,'tansig',WC2,BC2,'purelin', ...
                    WM1,BM1,'tansig',WM2,BM2,'purelin',Pc1,Pc2,TMh,eSIGN,n,DELAY,tpc);
end

if  n > tf/dt+1
    D = sqrt((w0*x(1))^2 + (GdP_Coef*deltaP)^2);
    if D < Dr
        lr = min(lr_max,lr*(Dr/D));
        tpc = [tpc(1:2),lr];
    end
end

dw0 = alpha*dwrN(n);  dp0 = gamma*dPg(n);
dw1 = alpha*dwrN(n-1);  dp1 = gamma*dPg(n-1);
det0 = dEtNs(n);  det1 = dEtNs(n-1);
P1 = [dw0;dw1;dp0;dp1];
u = simuff(P1,WC1,BC1,'tansig',WC2,BC2,'purelin');
u = min(vsmax,max(vsmin,G*u));
G = G0;
```

```
function [w1,w2] = tbpxNNC_N2(w1,b1,f1,w2,b2,f2,w3,b3,f3,w4,b4,f4,p1,p2,t,eSIGN,n,DELAY,tp)
% function to train NNC (Neural Network Controller) w/backpropagation.
% a revesion of tbpx function in MATLAB neural network toolbox

global u_NNC d2_A PNC vsmax vsmin
ecoeff = eSIGN*0.1/PNC;
if nargin < 18,error('Not enough arguments.');end

% TRAINING PARAMETERS
if nargin == 18, tp = []; end
tp = nndef(tp,[10 1e-6 0.1 1.0 0.7 0.9 1.04]);
me = tp(1); eg = tp(2); lr = tp(3); im = tp(4); dm = tp(5); mc = tp(6); er = tp(7);
W3 = w3(:,1);

df1 = feval(f1,'delta');
df2 = feval(f2,'delta');
df3 = feval(f3,'delta');
df4 = feval(f4,'delta');

dw1 = w1*0; dw2 = w2*0;
MC = 0;
```

```
% PRESENTATION PHASE
a1 = feval(f1,w1*p1,b1);
a2 = max(vsmin,min(vsmax,feval(f2,w2*a1,b2)));
u_NNC = a2;
A2_in = AVR_NNC_F(n,DELAY);
 A2 = [A2_in;p2];
a3 = feval(f3,w3*A2,b3);
a4 = feval(f4,w4*a3,b4);
err = a4 - t; e = ecoeff*err;
SSE = sumsqr(err);

 % BACKPROPAGATION PHASE
 d4 = feval(df4,a4,e);
 d3 = feval(df3,a3,d4,w4);
 d2 = feval(df2,a2,d3,W3);
 d2_A(n,:) = d2;
 D2 = AVR_NNC_B2(n,DELAY);
 d1 = feval(df1,a1,D2,w2);

for i=1:me

 % CHECK PHASE
 if SSE < eg | abs(a2) == vsmax ; break, end

 % LEARNING PHASE
  dw1 = learnwbm(p1,d1,lr,MC,dw1);
  dw2 = learnwbm(a1,D2,lr,MC,dw2);
 MC = mc;
 new_w1 = w1 + dw1;
 new_w2 = w2 + dw2;

 % PRESENTATION PHASE
 new_a1 = feval(f1,new_w1*p1,b1);
 new_a2 = max(vsmin,min(vsmax,feval(f2,new_w2*new_a1,b2)));
u_NNC = new_a2;
A2_in = AVR_NNC_F(n,DELAY);
new_A2 = [A2_in;p2];
  new_a3 = feval(f3,w3*new_A2,b3);
  new_a4 = feval(f4,w4*new_a3,b4);
  new_err = new_a4 - t;

  new_e = ecoeff*new_err;
  new_SSE = sumsqr(new_err);

  % MOMENTUM & ADAPTIVE LEARNING RATE PHASE
  if new_SSE > SSE*er
    lr = lr * dm;
    MC = 0;
  else
    if new_SSE < SSE
```

```
    lr = lr * im;
    end
   w1 = new_w1; a1 = new_a1;
   w2 = new_w2; a2 = new_a2;
   a3 = new_a3; a4 = new_a4;
   e = new_e; SSE = new_SSE;

  % BACKPROPAGATION PHASE
  d4 = feval(df4,a4,e);
  d3 = feval(df3,a3,d4,w4);
  d2 = feval(df2,a2,d3,W3);
  d2_A(n,:) = d2;
  D2 = AVR_NNC_B2(n,DELAY);
  d1 = feval(df1,a1,D2,w2);
  end
end
```

```
function D2 = AVR_NNC_B2(n,DELAY)
% function that solves the state space equations of the AVR in the backward path of the tbpxNNC program

global a b c KA d2_A D2 Nfinal DELTAt

% AVR parameters
TA=0.05; EFmax=5.0; EFmin=-2.0;
qk=0;

COL_LEN = size(d2_A,2);
kk1=zeros(4,COL_LEN);

if n > DELAY
    u_nnc = d2_A(n-DELAY,:);
else
    u_nnc = zeros(1,2);
end
VE = u_nnc;
Efd_nnc = 0;
for col=1:COL_LEN
    for index=1:4
        kk1(index,col)=a(index)*((KA*VE(col)-Efd_nnc)/TA - b(index)*qk);
        qk = qk + 3*kk1(index,col) - c(index)*(KA*VE(col)-Efd_nnc)/TA;
        Efd_nnc = Efd_nnc + kk1(index,col)*DELTAt;
    end

    if Efd_nnc<EFmin
        Efd_nnc=EFmin;
    elseif Efd_nnc>EFmax
        Efd_nnc=EFmax;
    end

    D2(col) = Efd_nnc;
```

```
end
```

```
function A2_in = AVR_NNC_F(n,DELAY)
% function that solves the state space equations of the AVR
% in forward path of the tbpxNNC program

global a b c G KA Rfd Xad Eta v1 Vref Efd0 u_NNC Efd dEFd_NNC_A Nfinal DELTAt

% AVR parameters
TA=0.05; TR=0.015; EFmax=5.0; EFmin=-2.0;
qv1=0;qk=0;
v1_NNC = v1;
Efd_NNC = Efd;
COL_LEN = length(u_NNC);
kv1=zeros(4,1); kk1=zeros(4,COL_LEN);

for index=1:4
    kv1(index)=a(index)*((Eta-v1_NNC)/TR-b(index)*qv1);
    qv1 = qv1 + 3*kv1(index) - c(index)*(Eta-v1_NNC)/TR;
    v1_NNC = v1_NNC + kv1(index)*DELTAt;
end

VE = Vref-v1_NNC+u_NNC;

for col=1:COL_LEN
    for index=1:4
        kk1(index,col)=a(index)*((KA*VE(col)-Efd_NNC)/TA - b(index)*qk);
        qk = qk + 3*kk1(index,col) - c(index)*(KA*VE(col)-Efd_NNC)/TA;
        Efd_NNC = Efd_NNC + kk1(index,col)*DELTAt;
    end

    if Efd_NNC<EFmin
        Efd_NNC=EFmin;
    elseif Efd_NNC>EFmax
        Efd_NNC=EFmax;
    end

    dEFd_NNC_A(n,col) = Efd_NNC-Efd0;

    if n > DELAY
        A2_in(col) = G*dEFd_NNC_A(n-DELAY,col);
    else
        A2_in(col) = 0;
    end
end
```

(c) AFPSSs

% wf_indaf.m %

```
% Response of the Cogenerator after occurence of a fault
% (two responses: system with conventional PSS; system with indirect AFPSS)

GLOB_init; simu_para; COG_data;
indafpss
C_fault

mainFIGS2
```

% indafpss.m %

```
% Response of the Cogenerator after occurence of a fault
% system with an indirect adaptive fuzzy PSS;

NIter = 2
Strt = 2;

for jjj = Strt : NIter
    Iteration = jjj
    clear tout,  clear yout, clear dwrl, clear usl, clear Etas, clear J
    global Efd0 Efd v10 Vref Eta Et0 Pe Q DELTAt tol x0 Pm SPdevp XX CNT count
    main1
    ytdot = 0; Ym0 = 0;
    XX = [0; 0];
    CNT = 100; count = 1;

    % Initializing the output array
    yout0 = [w0*x0(1); x0(2);x0(3);x0(4);x0(5);x0(6); Eta; Efd0; Pe; Q; u_cont; ytdot; Ym0];

    if jjj == Strt
        u_cont = ind_afc0;
    end

    SPdevp = w0*yout0(1);
    if jjj == 1
        [tout,yout,J,k] = ind_lf(k0,yout0,J0,t0,Tf,Efd0,Vref,DELTAt);
    else
        [tout,yout,J,k] = ind_l(k0,yout0,J0,t0,Tf,Efd0,Vref,DELTAt);
    end

    main2

    if jjj == 1
        [tout,yout,J,k] = ind_lf(k0,yout0,J0,t0,Tf,Efd0,Vref,DELTAt);
    else
        [tout,yout,J,k] = ind_l(k0,yout0,J0,t0,Tf,Efd0,Vref,DELTAt);
    end
```

```
    yout = [youtp; yout(k0:length(yout),:)];
    tout = [toutp, tout(k0:length(tout))];

    if jjj == 1
        dwrf = yout(:,1); deltaf = yout(:,2); usf = yout(:,11); Ytdf = yout(:,12); Pesf = yout(:,9);
        Jf = J
    else
        dwrI = yout(:,1); deltaI = yout(:,2); usI = yout(:,11); Ytdaf = yout(:,12); Pesaf = yout(:,9); Ym = yout(:,13);
        JafI = J
    end
end
```

% ind_lf.m %

```
function [tout,yout,J,k] = ind_lf(k0,yout0,J0,t0,Tf,Efd0,Vref,DELTAt)
% function that gives the output of the synchronous machine with the fixed initial fuzzy controller
% for one sampling period

global w0 u_cont Eta Pe Q x0 v10 Efd normTHta THta tf xm SPdevp

t = t0; J = J0; x = x0; k = k0;
tout(k) = t; yout(k,:) = yout0.';
yddotp = 0;

while t <= Tf
    u_cont = ind_fixed;              % fixed initial fuzzy controller
    [t,v1,Efd,Flag] = trns_avr(t0,Efd0,Vref,u_cont);
    if Flag == 1
        v10 = v1;
        Efd0 = Efd;
        clear t;
        delPp = - Pe;
        [t,x,Flag] = onest_de('sync_mST', t0, x0);
        if Flag == 1
            delP = - Pe;
            yddot = (delP-delPp)/DELTAt;
            ytdot = (yddot-yddotp)/DELTAt;
            yddotp = yddot;
            SPdevp = w0*x0(1);
            x0 = x; t0 = t;
            YY = [w0*x(1); x(2); x(3); x(4); x(5); x(6); Eta; Efd; Pe; Q; u_cont; ytdot; xm(1)];
            k = k+1;
            tout(k) = t;
            yout(k,:) = YY.';
            indf = abs(w0*x(1))*(t-tf);
            J = J + indf;
        end
    end
end
```

% ind_l.m %

```
function [tout,yout,J,k] = ind_l(k0,yout0,J0,t0,Tf,Efd0,Vref,DELTAt)
% function that gives the output of the synchronous machine with the indirect adaptive fuzzy controller
% for one sampling period

global w0 u_cont Eta Pe Q x0 v10 x Efd THtaf THtag tf xm SPdevp

t = t0; J = J0; x = x0; k = k0;
tout(k) = t; yout(k,:) = yout0.';
yddotp = 0;

while t <= Tf
    % Indirect adaptive fuzzy controller
    u_cont = ind_afc;
    [t,v1,Efd,Flag] = trns_avr(t0,Efd0,Vref,u_cont);
    if Flag == 1
        v10 = v1; Efd0 = Efd;
        clear t;
        delPp = - Pe;
        [t,x,Flag] = onest_de('sync_mST', t0, x0);
        if Flag == 1
            delP = - Pe;
            yddot = (delP-delPp)/DELTAt;
            ytdot = (yddot-yddotp)/DELTAt;
            yddotp = yddot;
            SPdevp = w0*x0(1);
            x0 = x; t0 = t;
            YY = [w0*x(1); x(2); x(3); x(4); x(5); x(6); Eta; Efd; Pe; Q; u_cont; ytdot; xm(1)];
            k = k+1;
            tout(k) = t;
            yout(k,:) = YY.';
            indf = abs(w0*x(1))*t;
            J = J + indf;
        end
    end
end
```

% ind_fixed.m %

```
% indirect adaptive fuzzy controller without adaptation

function u_cont = ind_fixed()
global Kd gAmmA1 gAmmA2 K Mf Mg M1 M2 Xbar1 Xbar2 Wmid1_1 Wmid1_2 Wmid2_1 Wmid2_2
global w1 wSt1 wEnd1 w2 wSt2 wEnd2 Mx1 Mx2 THtaf THtag xm xm0 w0 Eta Pm Pe Q Pn bc n Epsilon Eta Et0
global t0 v10 Efd0 Vref x0 DELTAt tol yout0 u_cont vsmin vsmax Eb
global SpCoef SPdevp dPCoef dPmCoef CoefXX margXX CNT XX count

count = count + 1;
b = bc(n);
```

```
SPdev = w0*x0(1);
deltaP = Pm - Pe;
XXp = XX;
XX = [SPdev; deltaP]; normXX = norm(XX,'fro');

if normXX ~= 0
    if Eta < 0.7*Et0
        xm = [SPdev; deltaP];
        xm0 = xm;
    else
        beneath = 1; % If equal to 1 the predictor will be the same as the one used for the ANNPSS
                % if equal to 2 the predictor will be based on the mass-spring-damper system
        if beneath == 1
            xm1 = SpCoef*SPdev - dPCoef*deltaP;
            xm = [xm1; dPmCoef*deltaP];
        else
            [t,xm1] = onest_deO('MsDo', 0, XXp, DELTAt); % Initial condition changes as the input changes
            xm = [xm1; dPmCoef*deltaP];
        end

        if rem(count,CNT) == 0
            Xout = XX'
            Xdesired = xm'
        end
    end

    e = XX - xm;
    M = M1*M2;
    PRODUCT = zeros(M,1);
    SUMMATION2 = 0;
    l = 0;
    for l1 = 1:M1
        SUMMATION1 = 0;
        xBAR = Xbar1(l1);
        if l1 == 1
            MUE1(l1) = mue_edge(XX(1),wSt1,xBAR);
        elseif l1 == M1
            MUE1(l1) = mue_edge(XX(1),wEnd1,xBAR);
        elseif l1 == (M1+1)/2
            MUE1(l1) = mue_mdle(XX(1),Wmid1_1,xBAR);
        else
            MUE1(l1) = mue_mdle(XX(1),Wmid1_2,xBAR);
        end

        for l2 = 1:M2
            xBAR = Xbar2(l2);
            if l2 == 1
                MUE2(l2) = mue_edge(XX(2),wSt2,xBAR);
            elseif l2 == M2
                MUE2(l2) = mue_edge(XX(2),wEnd2,xBAR);
            elseif l2 == (M2+1)/2
```

```
            MUE2(l2) = mue_mdle(XX(2),Wmid2_1,xBAR);
        else
            MUE2(l2) = mue_mdle(XX(2),Wmid2_2,xBAR);
        end
        l = l + 1;
        PRODUCT(l) = MUE1(l1)*MUE2(l2);
        SUMMATION1 = SUMMATION1 + PRODUCT(l);
    end
    SUMMATION2 = SUMMATION2 + SUMMATION1;
end
clear MUE1, clear MUE2

l = 0;
ksi = zeros(M,1);
for l1 = 1:M1
    for l2 = 1:M2
        l = l+1;
        ksi(l) = PRODUCT(l)/SUMMATION2;
    end
end

ksi0 = ksi;

clear PRODUCT
clear SUMMATION1, clear SUMMATION2

Fhat = THtaf'*ksi;
Ghat = THtag'*ksi;

u_cont = (-Fhat - K'*e)/Ghat;
if u_cont > vsmax
    u_cont = vsmax;
elseif u_cont < vsmin
    u_cont = vsmin;
end

clear ksi

else
    xm = [0; 0];
end
```

% ind_afc0.m %

```
% Initializing the indirect adaptive fuzzy controller
% Initializing parameters for estimating F(z) and G(z)

function u_cont = ind_afc0()

global Fhat Ghat Kd gAmmAf gAmmAg K Mf Mg M1 M2 Xbar1 Xbar2 Wmid1_1 Wmid1_2 Wmid2_1 Wmid2_2
global w1 wSt1 wEnd1 w2 wSt2 wEnd2 Mx1 Mx2 THtaf THtag THtaf0 THtag0 ksi0
```

```
global w0 x0 Pm Pe Pn bc n THtaHf THtaLg Eb Eta Et0 u_cont vsmin vsmax
global fU gU gL P VV DELTAt SpCoef dPCoef dPmCoef CoefXX margXX CNT XX count xm

count = count + 1;
gAmmAf = 2.0; gAmmAg = 20.0;
tol = 1e-3;
Mf = 3; Mg = 100;
THtaHf = 4; THtaLg = 0.5;
fU = 10; gU = 100; gL = 0.5; VV = 100;   % For suprvisory control
K1 = 1; K2 = 4; K = [K2; K1];

SPdev = w0*x0(1);
deltaP = Pm - Pe;

XXp = XX;
XX = [SPdev; deltaP]; normXX = norm(XX,'fro');

if Eta < 0.7*Et0
    xm = [SPdev; deltaP];
    xm0 = xm;
else
    beneath = 1;   % 1 the same predictor as for ANNPSS
                   % 2 the predictor based on mass-spring-damper system
    if beneath == 1
        xm1 = SpCoef*SPdev - dPCoef*deltaP;
        xm = [xm1; dPmCoef*deltaP];
    else
        [t,xm1] = onest_deO('MsDo', 0, XXp, DELTAt); % Initial conditions change with the input
        xm = [xm1; dPmCoef*deltaP];
    end

    if rem(count,CNT) == 0
        Xout = XX'
        Xdesired = xm'
    end
end

e = XX - xm;

n = 2;
V = 1.0*ones(1,n);
Q = diag(V);
LAMBDAc = [0 1; -K2 -K1];
P = lyap(LAMBDAc',Q);
bc = [0; 1];
Pn = P*bc;

InMemb

% Initialize the adjustable parameters
beneath = 0;
```

```
if beneath == 1
    THtaf(1)=-1; THtaf(2)=-0.8; THtaf(3)=0; THtaf(4)=0.1; THtaf(5)=0.2;
    THtaf(6)=-0.8; THtaf(7)=-0.6; THtaf(8)=0; THtaf(9)=0.3; THtaf(10)=0.4;
    THtaf(11)=-0.6; THtaf(12)=-0.45; THtaf(13)=0; THtaf(14)=0.45; THtaf(15)=0.6;
    THtaf(16)=-0.4; THtaf(17)=-0.3; THtaf(18)=0; THtaf(19)=0.6; THtaf(20)=0.8;
    THtaf(21)=-0.2; THtaf(22)=-0.1; THtaf(23)=0; THtaf(24)=0.8; THtaf(25)=1;
    THtaf0 = 0.05*THtaf';
else
    THtaf0 = zeros(25,1);
end

THtag(1)=-0.2; THtag(2)=-0.4; THtag(3)=-0.6; THtag(4)=-0.4; THtag(5)=-0.2;
THtag(6)=-0.4; THtag(7)=-0.6; THtag(8)=-0.8; THtag(9)=-0.6; THtag(10)=-0.4;
THtag(11)=-0.6; THtag(12)=-0.8; THtag(13)=-1; THtag(14)=-0.8; THtag(15)=-0.6;
THtag(16)=-0.4; THtag(17)=-0.6; THtag(18)=-0.8; THtag(19)=-0.6; THtag(20)=-0.4;
THtag(21)=-0.2; THtag(22)=-0.4; THtag(23)=-0.6; THtag(24)=-0.4; THtag(25)=-0.2;
THtag0 = 10*THtag';

THtaf = THtaf0; THtag = THtag0;

save matlab/final/AdF_pss/indirect/tempf/THtaf THtaf
save matlab/final/AdF_pss/indirect/tempf/THtag THtag

normTHtaf=norm(THtaf,'fro')
normTHtag=norm(THtag,'fro')

% Calculate initial Fhat and Ghat
Fhat = THtaf'*ksi
Ghat = THtag'*ksi

clear ksi

u_cont = (-Fhat - K'*e)/Ghat
if u_cont > vsmax
    u_cont = vsmax;
elseif u_cont < vsmin
    u_cont = vsmin;
end
```

% InMemb.m %

```
% Input membership functions for AFPSS's

Mx1 = 1.6;  % Define ranges of the fuzzy membership functions for input 1
Mx2 = 0.3;  % Define ranges of the fuzzy membership functions for input 2
M1 = 5; M2 = 5;
W_inv = 3;
COEFtemp = 2.0;  % Defining maximum range of membership functions graphs
coefT1 = 1.6; % Defining centre of gravity of other m.f. for speed deviation
coefT2 = 1.2; % Defining centre of gravity of other m.f. for acceleration
coefTE1 = 1.2;  % Defining centre of gravity of the edges m.f. for sp. dev.
coefTE2 = 1.0;  % Defining centre of gravity of the edges m.f. for acc.
```

```
w1 = W_inv/Mx1; wSt1 = 5*w1; wEnd1 = -wSt1;
Wmid1_1 = 2.6*w1; % width of zero membership function (bigger number makes the m.f. narrower)
Wmid1_2 = 0.8*w1;

for l = 1 : M1
    if l == 1
        Xbar1(l) = -coefTE1*Mx1;
    elseif l == M1
        Xbar1(l) = coefTE1*Mx1;
  else
        Xbar1(l) = coefT1*(2*l-(M1+1))*Mx1/(M1-1);
  end
end

x_A1 = [-COEFtemp*Mx1:0.02:COEFtemp*Mx1];

I = 2*COEFtemp*Mx1/0.02+1;
for i = 1:I
    xi1 = x_A1(i);
    for j = 1:M1
        if j == 1
            mue1(j,i) = mue_edge(xi1,wSt1,Xbar1(j));
        elseif j == M1
            mue1(j,i) = mue_edge(xi1,wEnd1,Xbar1(j));
        elseif j == (M1+1)/2
            mue1(j,i) = mue_mdle(xi1,Wmid1_1,Xbar1(j));
        else
            mue1(j,i) = mue_mdle(xi1,Wmid1_2,Xbar1(j));
        end
    end
end

figure(1); clf
for j = 1 :M1
    plot(x_A1,mue1(j,:))
    hold on
end
axis([-COEFtemp*Mx1 COEFtemp*Mx1 0 1]);
clear x_A1, clear mue1, clear xi1

w2 = W_inv/Mx2;
wSt2 = 8*w2;
wEnd2 = -wSt2;
Wmid2_1 = 20*w2;
Wmid2_2 = 8*w2;

for l = 1 : M2
    if l == 1
        Xbar2(l) = -coefTE2*Mx2;
    elseif l == M2
        Xbar2(l) = coefTE2*Mx2;
```

```
    else
        Xbar2(l) = coefT2*(2*l-(M2+1))*Mx2/(M2-1);
    end
end

x_A2 = [-COEFtemp*Mx2:0.001:COEFtemp*Mx2];
I = 2*COEFtemp*Mx2/0.001+1;
for i = 1:I
    xi2 = x_A2(i);
    for j = 1:M2
        if j == 1
            mue2(j,i) = mue_edge(xi2,wSt2,Xbar2(j));
        elseif j == M2
            mue2(j,i) = mue_edge(xi2,wEnd2,Xbar2(j));
        elseif j == (M2+1)/2
            mue2(j,i) = mue_mdle(xi2,Wmid2_1,Xbar2(j));
        else
            mue2(j,i) = mue_mdle(xi2,Wmid2_2,Xbar2(j));
        end
    end
end
figure(2); clf
for j = 1 :M2
    plot(x_A2,mue2(j,:))
    hold on
end
axis([-COEFtemp*Mx2 COEFtemp*Mx2 0 1]);
clear x_A2, clear mue2, clear xi2

% Construct the fuzzy basis functions

M = M1*M2;
PRODUCT = zeros(M,1);
SUMMATION2 = 0;
l = 0;
for l1 = 1:M1
    SUMMATION1 = 0;
    xBAR = Xbar1(l1);
    if l1 == 1
        MUE1(l1) = mue_edge(XX(1),wSt1,xBAR);
    elseif l1 == M1
        MUE1(l1) = mue_edge(XX(1),wEnd1,xBAR);
    elseif l1 == (M1+1)/2
        MUE1(l1) = mue_mdle(XX(1),Wmid1_1,xBAR);
    else
        MUE1(l1) = mue_mdle(XX(1),Wmid1_2,xBAR);
    end

    for l2 = 1:M2
        xBAR = Xbar2(l2);
        if l2 == 1
```

```
            MUE2(l2) = mue_edge(XX(2),wSt2,xBAR);
        elseif l2 == M2
            MUE2(l2) = mue_edge(XX(2),wEnd2,xBAR);
        elseif l2 == (M2+1)/2
            MUE2(l2) = mue_mdle(XX(2),Wmid2_1,xBAR);
        else
            MUE2(l2) = mue_mdle(XX(2),Wmid2_2,xBAR);
        end
        l = l + 1;
        PRODUCT(l) = MUE1(l1)*MUE2(l2);
        SUMMATION1 = SUMMATION1 + PRODUCT(l);
    end
    SUMMATION2 = SUMMATION2 + SUMMATION1;
end

clear MUE1, clear MUE2

l = 0;
ksi = zeros(M,1);
for l1 = 1:M1
    for l2 = 1:M2
        l = l+1;
        ksi(l) = PRODUCT(l)/SUMMATION2;
    end
end

ksi0 = ksi;

clear PRODUCT
clear SUMMATION1, clear SUMMATION2
```

```
% function mue_mdle(x,w,xBAR,Mx)
% xBAR is the centre of the membership function
% The width of the membership function is reversely proportional to w

function y = mue_mdle(x,w,xBAR)
% this function is maximum (=1) at x = xBAR

y = exp(-w*(x-xBAR)^2);
```

```
% function mue_edge(x,w,xBAR)

function y = mue_edge(x,w,xBAR)
% width of the membership function is reversely proportional to |w|
% at xBAR the value of the membership function is 0.5

y = 1/(1+exp(w*(x-xBAR)));
```

```
% This function is a mass-spring-damper system
```

```
%  y + lambda1*y + lambda2*y = 0
% y = x(1)

function y = MsDo(t,x)
lambda1 = 5;  % For Latrobe Cogenerator
lambda2 = 20;  % For Latrobe Cogenerator
%lambda1 = 5;  % for Crown Cogenerator
%lambda2 = 50; % for Crown Cogenerator

xdot = zeros(2,1);
xdot(1) = x(2);
xdot(2) = -lambda2*x(1)-lambda1*x(2);

y = xdot(1);
```

% ind_afc.m %

```
% On_line adaptation of the indirect adaptive fuzzy controller

function u_cont = ind_afc()

global Kd gAmmAf gAmmAg K Mf Mg M1 M2 Xbar1 Xbar2 Wmid1_1 Wmid1_2 Wmid2_1 Wmid2_2
global w1 wSt1 wEnd1 w2 wSt2 wEnd2 Mx1 Mx2 xm xm0 w0 Eta Pm Pe Q Pn bc n THtaHf THtaLg Eta Et0
global t0 v10 Efd0 Vref x0 DELTAt tol yout0 u_cont vsmin vsmax Eb fU gU gL P VV DELTAt
global H SpCoef SPdevp dPCoef dPmCoef CoefXX margXX CNT XX count

count = count + 1;

load matlab/final/AdF_pss/indirect/tempf/THtaf0
load matlab/final/AdF_pss/indirect/tempf/THtaf
load matlab/final/AdF_pss/indirect/tempf/THtag

b = bc(n);
SPdev = w0*x0(1);
deltaP = Pm - Pe;

XXp = XX;
XX = [SPdev; deltaP];
normXX = norm(XX,'fro');

if normXX ~= 0
    if Eta < 0.7*Et0
        xm = [SPdev; deltaP];
        xm0 = xm;
    else
        beneath = 1;  % 1 the predictor the same as the one used for ANNPSS
                % 2 the predictor based on the mass-spring-damper system
        if beneath == 1
            xm1 = SpCoef*SPdev - dPCoef*deltaP;
            xm = [xm1; dPmCoef*deltaP];
```

```
    else
        [t,xm1] = onest_deO('MsDo', 0, XXp, DELTAt);
        xm = [xm1; dPmCoef*deltaP];
    end
    beneath = 0;
    if beneath == 1
        if rem(count,CNT) == 0
            Xout = XX'
            Xdesired = xm'
        end
    end
end
e = XX - xm;

M = M1*M2;
PRODUCT = zeros(M,1);
SUMMATION2 = 0;
l = 0;
for l1 = 1:M1
    SUMMATION1 = 0;
    xBAR = Xbar1(l1);
    if l1 == 1
        MUE1(l1) = mue_edge(XX(1),wSt1,xBAR);
    elseif l1 == M1
        MUE1(l1) = mue_edge(XX(1),wEnd1,xBAR);
    elseif l1 == (M1+1)/2
        MUE1(l1) = mue_mdle(XX(1),Wmid1_1,xBAR);
    else
        MUE1(l1) = mue_mdle(XX(1),Wmid1_2,xBAR);
    end

    for l2 = 1:M2
        xBAR = Xbar2(l2);
        if l2 == 1
            MUE2(l2) = mue_edge(XX(2),wSt2,xBAR);
        elseif l2 == M2
            MUE2(l2) = mue_edge(XX(2),wEnd2,xBAR);
        elseif l2 == (M2+1)/2
            MUE2(l2) = mue_mdle(XX(2),Wmid2_1,xBAR);
        else
            MUE2(l2) = mue_mdle(XX(2),Wmid2_2,xBAR);
        end
        l = l + 1;
        PRODUCT(l) = MUE1(l1)*MUE2(l2);
        SUMMATION1 = SUMMATION1 + PRODUCT(l);
    end
    SUMMATION2 = SUMMATION2 + SUMMATION1;
end

clear MUE1, clear MUE2
```

```
l = 0;
ksi = zeros(M,1);
for l1 = 1:M1
    for l2 = 1:M2
        l = l+1;
        ksi(l) = PRODUCT(l)/SUMMATION2;
    end
end

ksi0 = ksi;
clear PRODUCT
clear SUMMATION1, clear SUMMATION2

Fhat = THtaf'*ksi;
Ghat = THtag'*ksi;
uc = (-Fhat + K'*e)/Ghat;

if normXX > 0.05
    Gamma1N = gAmmAf;
    Gamma2N = gAmmAg;
    normTHtaf = norm(THtaf,'fro');

    if  normTHtaf < Mf | (normTHtaf >= Mf & e'*Pn*THtaf'*ksi < 0)
        delTHtaf = Gamma1N*e'*Pn*b*ksi;
    elseif  normTHtaf >= Mf & e'*Pn*THtaf'*ksi >= 0
        temp = THtaf*THtaf'*ksi/normTHtaf^2;
        delTHtaf =(Gamma1N*e'*Pn*b*ksi-Gamma1N*e'*Pn*b*temp);
    end

    clear temp
    THtaf = THtaf + delTHtaf;
    clear delTHtaf;
    EQpoint = (M+1)/2;
    THtaf(EQpoint) = 0;

    % If necessary to do the following put Beneath = 1
    Beneath = 0;
    if Beneath == 1
        I = find(abs(THtaf) > THtaHf);
        for m = 1:length(I)
            THtaf(I(m)) = sign(THtaf(I(m)))*THtaHf;
        end
    end

    normTHtag = norm(THtag,'fro');
    if normTHtag < Mg | (normTHtag>=Mg & e'*Pn*THtag'*ksi*u_cont < 0)
        delTHtag = Gamma2N*e'*Pn*b*ksi*sign(u_cont);
    elseif  normTHtag >= Mg & e'*Pn*THtag'*ksi*u_cont >= 0
        temp = THtag*THtag'*ksi*sign(u_cont)/normTHtag^2;
        delTHtag =(Gamma2N*e'*Pn*b*ksi*sign(u_cont)-Gamma2N*e'*Pn*b*temp);
    end
```

```
        clear temp
        THtag = THtag + delTHtag;
        clear delTHtag;

        beneath = 0;
        if beneath == 1
            I = find(abs(THtag) < THtaLg);
            for m = 1:length(I)
                THtag(I(m)) = sign(THtag(I(m)))*THtaLg;
            end
        end
    else
        THtaf = THtaf0;
    end

    Fhat = THtaf'*ksi;
    Ghat = THtag'*ksi;
    uc = (-Fhat - K'*e)/Ghat;

    % Supervisory control
    Ve = 0.5*e'*P*e;
    if Ve > VV
        us = sign(e'*Pn)*(abs(Fhat)+fU+abs(Ghat*uc)+abs(gU*uc))/gL
    else
        us = 0;
    end

    if normXX > 0.001
        u_cont = uc + us;
    else
        u_cont = 0;
    end

    if u_cont > vsmax
        u_cont = vsmax;
    elseif u_cont < vsmin
        u_cont = vsmin;
    end
    clear ksi
else
    xm = [0; 0];
end
save matlab/final/AdF_pss/indirect/tempf/THtaf THtaf
save matlab/final/AdF_pss/indirect/tempf/THtag THtag
```

% wFaF_PSS.m %

```
% Response of the Cogenerator after occurence of a fault
%(four responses: system with a direct AFPSS; system with conventional PSS;
```

```
% system with rule-based fuzzy PSS; system with neuro-fuzzy PSS)

GLOB_init; simu_para; COG_data
adfpss
deltaD = deltaD*180/pi;

C_fault
deltac = deltac*180/pi;

spCOEF = 0.2; accCOEF = 1.1;  % best for Pt0=0.9, Fp=0.9, Xe=0.15
uCOEF = 0.5;

F_fault
deltaf = deltaf*180/pi;

FN_fault
deltafn = deltafn*180/pi;

LW = 1.6;
main_figs4   % For CPSS, FPSS, NFPSS, DAFPSS
```

% adfpss.m %

```
% Response of the Cogenerator after occurence of a fault (system with a direct AFPSS);

NIter = 2   % For the initial FPSS this should be equal to 1
            % For the on-line adaptation it should be 2

Strt = 2;
for jjj = Strt : NIter
    Iteration = jjj
    clear tout,  clear yout
    clear dwrD, clear usD, clear VtD; clear Etas
    clear J

    global Efd0 Efd v10 Vref Eta Et0 Pe Q DELTAt Pe_ba x0 Pm XX count
    count = 1;
    main1
    XX = [0; 0];
    Ym0 = 0;
    PeDot = 0;

    % Initializing the output array
    yout0 = [w0*x0(1); x0(2);x0(3);x0(4);x0(5);x0(6); Eta; Efd0; Pe; PeDot; Q; u_cont; Ym0];

    if jjj == Strt
        u_cont = adpt_fc0;
    end

    if jjj == 1
        [tout,yout,J,k] = mloopf_ST(k0,yout0,J0,t0,Tf,Efd0,Vref,DELTAt);
```

```
    else
        [tout,yout,J,k] = mloop_ST(k0,yout0,J0,t0,Tf,Efd0,Vref,DELTAt);
    end

    main2
    if jjj == 1
        [tout,yout,J,k] = mloopf_ST(k0,yout0,J0,t0,Tf,Efd0,Vref,DELTAt);
    else
        [tout,yout,J,k] = mloop_ST(k0,yout0,J0,t0,Tf,Efd0,Vref,DELTAt);
    end

    yout = [youtp; yout(k0:length(yout),:)];
    tout = [toutp, tout(k0:length(tout))];

    if jjj == 1
        dwrf = yout(:,1); deltaf = yout(:,2); Ym = yout(:,13); PEdot = yout(:,10); Vtf = yout(:,7); usf = yout(:,12);
        Jf = J
    else
        dwrD = yout(:,1); deltaD = yout(:,2); Ym = yout(:,13); PEdot = yout(:,10); VtD = yout(:,7); usD = yout(:,12);
        JafD = J
    end

end
```

```
% mloopf_ST.m
% The main loop consisting of the AVR & exciter and the synchronous machine with a fixed initial fuzzy controller
    % use:
    % u_cont = fixed_fc;
    % in the main loop (see mfile FUZloop_ST)
```

```
% mloop_ST.m
% The main loop consisting of the AVR & exciter and the synchronous machine
% with the direct AFPSS with on-line adaptation
    % use:
    % u_cont = adapt_fc;
    % in the main loop (see mfile FUZloop_ST)
```

% adpt_fc0.m %

```
% Initializing the direct adaptive fuzzy controller

function u_cont = adpt_fc0()

global u_cont Kd gAmmA MTHta
global M1 M2 Xbar1 Xbar2 Wmid1_1 Wmid1_2 Wmid2_1 Wmid2_2 w1 wSt1 wEnd1 w2 wSt2 wEnd2 Mx1 Mx2
global THta THta0 THtaR ksi0 w0 x0 Pm Pe P Eta Et0 vsmin vsmax K bL fU VV SpCoef dPCoef dPmCoef

gAmmA = 1.0;
```

```
MTHta = 3;
K1 = 1; K2 = 4; K = [K2; K1];
bL = 0.2; fU = 5; VV = 5;

SPdev = w0*x0(1);
deltaP = Pm - Pe - Kd*SPdev;

XX = [SPdev; deltaP];
n = 2;
V = 1.0*ones(1,n);
Q = diag(V);
LAMBDAc = [0 1; -K2 -K1];
P = lyap(LAMBDAc',Q);

InMemb

% Initialize the adjustable parameters

%THtaR = 0.2*(rand(M,1)-0.5);

THta1_5 = [-1; -0.6; -0.2; 0; 0.1];
THta6_10 = [-0.8; -0.4; -0.1; 0.1; 0.25];
THta11_15 = [-0.5; -0.25; 0; 0.25; 0.5];
THta16_20 = [-0.25; -0.1; 0.1; 0.4; 0.8];
THta21_25 = [-0.1; 0; 0.2; 0.6; 1];
THta0 = 0.3*[THta1_5;THta6_10;THta11_15;THta16_20;THta21_25];

clear THta1_5; clear THta6_10; clear THta11_15;
clear THta16_20; clear THta21_25

THta = 1.0*THta0;
%THta = 1.0*THtaR;
save matlab/AdFuz/AdF_PSS/direct/tempf/THta THta

normTHta=norm(THta,'fro')

% Calculate the output of the controller
u_cont = THta'*ksi;
if u_cont > vsmax
    u_cont = vsmax;
elseif u_cont < vsmin
    u_cont = vsmin;
end

clear ksi
```

% fixed_fc.m %

```
% fixed fuzzy controller before adaptation
```

```
function u_cont = fixed_fc()

global Kd gAmmA MTHta M1 M2 Xbar1 Xbar2 Wmid1_1 Wmid1_2 Wmid2_1 Wmid2_2
global w1 wSt1 wEnd1 w2 wSt2 wEnd2 Mx1 Mx2 THta w0 Eta Pm Pe Q Pn Eta Et0
global t0 v10 Efd0 Vref u_cont x0 DELTAt tol yout0 vsmin vsmax Eb xm

SPdev = w0*x0(1);
deltaP = Pm - Pe - Kd*SPdev;

normTHta=norm(THta,'fro');
XX = [SPdev; deltaP];
normXX = norm(XX,'fro');

M = M1*M2;
PRODUCT = zeros(M,1);
SUMMATION2 = 0;
l = 0;
for l1 = 1:M1
    SUMMATION1 = 0;
    xBAR = Xbar1(l1);
   if l1 == 1
        MUE1(l1) = mue_edge(XX(1),wSt1,xBAR);
    elseif l1 == M1
        MUE1(l1) = mue_edge(XX(1),wEnd1,xBAR);
    elseif l1 == (M1+1)/2
        MUE1(l1) = mue_mdle(XX(1),Wmid1_1,xBAR);
    else
        MUE1(l1) = mue_mdle(XX(1),Wmid1_2,xBAR);
    end

    for l2 = 1:M2
        xBAR = Xbar2(l2);
        if l2 == 1
            MUE2(l2) = mue_edge(XX(2),wSt2,xBAR);
        elseif l2 == M2
            MUE2(l2) = mue_edge(XX(2),wEnd2,xBAR);
        elseif l2 == (M2+1)/2
            MUE2(l2) = mue_mdle(XX(2),Wmid2_1,xBAR);
        else
            MUE2(l2) = mue_mdle(XX(2),Wmid2_2,xBAR);
        end

        l = l + 1;
        PRODUCT(l) = MUE1(l1)*MUE2(l2);
        SUMMATION1 = SUMMATION1 + PRODUCT(l);
    end
    SUMMATION2 = SUMMATION2 + SUMMATION1;
end

clear MUE1, clear MUE2
```

```
l = 0;
ksi = zeros(M,1);
for l1 = 1:M1
    for l2 = 1:M2
        l = l+1;
        ksi(l) = PRODUCT(l)/SUMMATION2;
    end
end

clear PRODUCT, clear SUMMATION1, clear SUMMATION2

u_cont = THta'*ksi;

if u_cont > vsmax
    u_cont = vsmax;
elseif u_cont < vsmin
    u_cont = vsmin;
end

clear ksi
```

% adapt_fc.m %

```
% On_line adaptation of the direct adaptive fuzzy controller

function u_cont = adapt_fc()

global Kd gAmmA MTHta M1 M2 Xbar1 Xbar2 Wmid1_1 Wmid1_2 Wmid2_1 Wmid2_2
global w1 wSt1 wEnd1 w2 wSt2 wEnd2 Mx1 Mx2 THta w0 Eta Pm Pe Q P H Eta Et0 t0 v10 Vref x0
global DELTAt tol yout0 u_cont vsmin vsmax Eb x0 Rfd Xfd Xad R1d X1d R1q X1q R2q X2q Eb Xta
global Xd Xq Xdp Xqp Xddp Xqdp Xaqsp Xadsp Xaqsdp Xadsdp
global K bL fU VV xm xm0 SpCoef dPCoef dPmCoef count CNT NMB XX XXp XXX

load matlab/AdFuz/AdF_PSS/direct/tempf/THta0
load matlab/AdFuz/AdF_PSS/direct/tempf/THta

normTHta=norm(THta,'fro');
XXX = [XXX,XX];

if count > NMB
    XXp = XXX(:,count-NMB);
else
    XXp = XX;
end

SPdev = w0*x0(1);
deltaP = Pm - Pe;
XX = [SPdev; deltaP];
normXX = norm(XX,'fro');
Pn = P(:,2);
```

```
if normXX ~= 0
    if Eta < 0.7*Et0
        xm = XX;
        xm0 = xm;
    else
        beneath = 1;  % 1 the predictor the same as the one used for ANNPSS
                      % 2 the predictor based on the mass-spring-damper system
        if beneath == 1
            xm1 = SpCoef*SPdev - dPCoef*deltaP;
            xm = [xm1; dPmCoef*deltaP];
        else
            [t,xm1] = onest_deO('MsDo', 0, XXp, DELTAt);
            xm = [xm1; dPmCoef*deltaP];
        end
        beneath = 0;
    end
    e = XX - xm;
    M = M1*M2;
    PRODUCT = zeros(M,1);
    SUMMATION2 = 0;
    l = 0;

    for l1 = 1:M1
        SUMMATION1 = 0;
        xBAR = Xbar1(l1);
        if l1 == 1
            MUE1(l1) = mue_edge(XX(1),wSt1,xBAR);
        elseif l1 == M1
            MUE1(l1) = mue_edge(XX(1),wEnd1,xBAR);
        elseif l1 == (M1+1)/2
            MUE1(l1) = mue_mdle(XX(1),Wmid1_1,xBAR);
        else
            MUE1(l1) = mue_mdle(XX(1),Wmid1_2,xBAR);
        end

        for l2 = 1:M2
            xBAR = Xbar2(l2);
            if l2 == 1
                MUE2(l2) = mue_edge(XX(2),wSt2,xBAR);
            elseif l2 == M2
                MUE2(l2) = mue_edge(XX(2),wEnd2,xBAR);
            elseif l2 == (M2+1)/2
                MUE2(l2) = mue_mdle(XX(2),Wmid2_1,xBAR);
            else
                MUE2(l2) = mue_mdle(XX(2),Wmid2_2,xBAR);
            end
            l = l + 1;
            PRODUCT(l) = MUE1(l1)*MUE2(l2);
            SUMMATION1 = SUMMATION1 + PRODUCT(l);
        end
```

```
    SUMMATION2 = SUMMATION2 + SUMMATION1;
end
clear MUE1, clear MUE2

l = 0;
ksi = zeros(M,1);
for l1 = 1:M1
    for l2 = 1:M2
        l = l+1;
        ksi(l) = PRODUCT(l)/SUMMATION2;
    end
end

clear PRODUCT, clear SUMMATION1, clear SUMMATION2

uc = THta'*ksi;

if normXX > 0.05
    if uc > 0.015
        GammaN = gAmmA/(uc/0.015)^2;
    else
        GammaN = gAmmA;
    end

    if  normTHta < MTHta | (normTHta >= MTHta & e'*Pn*THta'*ksi < 0)
        delTHta = GammaN*e'*Pn*ksi;
    elseif  normTHta >= MTHta & e'*Pn*THta'*ksi >= 0
        temp = THta*THta';
        temp = temp*ksi/normTHta^2;
        delTHta =(GammaN*e'*Pn*ksi-GammaN*e'*Pn*temp);
    end
    clear temp

    THta = THta + delTHta;
    EQpoint = (M+1)/2; THta(EQpoint) = 0;

    clear delTHta;
    THtaa = THta;
else
    THta = THta0;
end
normTHta=norm(THta,'fro');

uc = THta'*ksi;

Ve = 0.5*e'*P*e;
if Ve > VV
    us = sign(e'*Pn)*(abs(uc)+(fU+abs(K'*e))/bL)
    u_cont = uc + us;
else
    u_cont = uc;
```

```
    end

    if normXX < 0.001
        u_cont = 0;
    end

    if u_cont > vsmax
        u_cont = vsmax;
    elseif u_cont < vsmin
        u_cont = vsmin;
    end
    clear ksi
else
    xm = [0; 0];
end

save matlab/AdFuz/AdF_PSS/direct/tempf/THta THta
```

Part III: Shaping figures

% main_figs4.m %

```
% Main figures for four responses used in the programs

figure(1);clf;
plot(tout,deltac,'b:',tout,deltaf,'g--',tout,deltafn,'r-.',tout,deltaD,'k');
xlabel('t, sec','FontSize',14);
ylabel('torque angle, degrees','FontSize',14);
grid

Y_Y = [deltac;deltaf;deltafn;deltaD];
fig_MAKEUP

figure(2);clf;
plot(tout,dwrc,'b:',tout,dwrf,'g--',tout,dwrfn,'r-.',tout,dwrD,'k');

xlabel('t, sec','FontSize',14);
ylabel('speed deviation, rad/sec','FontSize',14);
grid

Y_Y = [dwrc;dwrf;dwrfn;dwrD];
fig_MAKEUP

figure(3);clf;
plot(tout,usc,'b:',tout,usf,'g--',tout,usfn,'r-.',tout,usD,'k'); grid;
xlabel('t, sec','FontSize',14);
ylabel('stabilising signal, pu','FontSize',14);

Y_Y = [usc;usf;usfn;usD];
fig_MAKEUP

figure(4);clf;
plot(tout,Vtc,'b:',tout,Vtf,'g--',tout,Vtfn,'r-.',tout,VtD,'k'); grid;
xlabel('t, sec','FontSize',14);
ylabel('Terminal voltage, pu','FontSize',14);

Y_Y = [Vtc;Vtf;Vtfn;VtD];
fig_MAKEUP
```

```
% fig_MAKEUP.m
minY = min(Y_Y)-(max(Y_Y)-min(Y_Y))/10;
maxY = 1.2*max(Y_Y);
axis([0 5 minY maxY])
hh = get(gca,'children');
set(hh,'LineWidth',LW);
hh = get(gcf,'children');
set(hh(1),'FontSize',14);
P_g = (maxY-minY)/20;
P_G = P_g/2;
```

```
maxP = maxY - P_G;
minP = maxY - 5.5*P_g;
patch([3.3;3.3;4.8;4.8],[minP;maxP;maxP;minP],'w')
Y_P = maxP - P_g;
line([3.4;3.8],[Y_P;Y_P],'LineStyle',':','Color','b','LineWidth',LW)
text(3.9,Y_P,'CPSS','FontSize',16)
Y_P = Y_P - P_g;
line([3.4;3.8],[Y_P;Y_P],'LineStyle','--','Color','g','LineWidth',LW)
text(3.9,Y_P,'FPSS','FontSize',16)
Y_P = Y_P - P_g;
line([3.4;3.8],[Y_P;Y_P],'LineStyle','-.','Color','r','LineWidth',LW)
text(3.9,Y_P,'NFPSS','FontSize',16)
Y_P = Y_P - P_g;
line([3.4;3.8],[Y_P;Y_P],'LineStyle','-','Color','k','LineWidth',LW)
text(3.9,Y_P,'DAFPSS','FontSize',16)
```