THE UNIVERSITY OF CALGARY

Neural Adaptive Power System Stabilizer

by

Payman Shamsollahi

A DISSERTATION

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

CALGARY, ALBERTA

JULY, 1997

0-612-24562-4

Abstract

Power system stabilizers are responsible for enhancing the power system stability and for improving the dynamic performance of the system. An adaptive power system stabilizer using on-line trained neural networks is developed in this dissertation. The feed-forward multi-layer neural network along with the back-propagation algorithm in on-line mode is used to design the neural adaptive power system stabilizer (NAPSS) . The structure and training procedure of the proposed NAPSS are discussed.

The proposed NAPSS consists of an identifier to track and identify the non-linear plant in real-time and a controller to damp power plant oscillations. These two subnetworks are trained in each sampling period employing the on-line version of the back-propagation algorithm. The resulting NAPSS does not require any reference model or teacher and is trained directly based on output performance of the plant. It also does not need the internal states of the plant to be measured and just uses the output of the plant. The NAPSS is tested on a single-machine infinite-bus power system model for a variety of disturbances.

A multi-machine power system is used to evaluate the performance of the NAPSS in damping power system multi-mode oscillations. The effectiveness of the NAPSS in damping multi-mode oscillations and its self-coordination ability are also demonstrated. A Digital Signal Processor (DSP) board is employed to implement the NAPSS. The behavior of the NAPSS is then investigated using a physical model of a power system in the Power System Research Laboratory at the University of Calgary. Implementation steps and real-time test results are presented.

Acknowledgments

As with any undertaking, the four-year work which went into the completion of this dissertation could not have been done without the support and interest of a number of people.

My supervisor, Dr. O. P. Malik, must be especially acknowledged for his scholarly advice, constant guidance and support throughout the program. From him, I have learned a lot not only in the area of my specialty but also in the style of teaching and research.

The laboratory work was supported greatly by Mr. G. C. Hancock and Mr. P. Walsh. I offer warm thanks to our graduate secretary, Ms. E. Lok, for her swift replies to my numerous inquiries.

I am deeply grateful to all my dear friends who helped me in many ways during this endeavor.

My parents have always given me the encouragement to learn more and employ my knowledge in practical ways. I feel eternal gratitude to them for their lives they have sacrificed for me.

My special thanks must go to my wife Farnoosh who encouraged me through the challenges of this research.

To
My Parents and My Wife

Contents

List of Tables

List of Figures

List of Symbols

Symbol	Definition (page where first used/defined)
ABB	Asea Brown Boveri Co. (15)
AC	Alternating Current (6)
ADALINE	ADAptive LINear Element (19)
A/D	Analog to Digital converter (38)
AI	Artificial Intelligence (8)
ANC	Adaptive Neuro-Controller (44)
ANI	Adaptive Neuro-Identifier (44)
AVR	Automatic Voltage Regulator (15)
B_t	transmission line susceptance (168)
CMAC	Cerebellar Model Arithmetic Computer (19)
CPSS	Conventional Power System Stabilizer (6)
DARAM	Dual Access RAM (122)
D/A	Digital to Analog converter (38)
DC	Direct Current (114)
DMA	Direct Memory Access (120)
DSP	Digital Signal Processor/Processing (15)
$\Delta P_e(k)$	generator accelerating power at time step k (75)
$\Delta P_{ed}(k)$	desired generator accelerating power at time step k (100)
$\Delta \hat{P}_e(k)$	predicted generator accelerating power at time step k (97)
$\Delta P_E(s)$	Laplace transform of the accelerating power (105)
$\Delta\omega_d(k)$	desired speed deviation at time step k (53)
$\Delta\hat{\omega}(k)$	generator predicted speed deviation at time step k (60)
$\Delta\omega(k)$	generator speed deviation at time step k (60)

$\Delta w_{ji}(n)$	correction applied to the weight $w_{ji}(n)$ (31)
E_p	total squared error for pattern p (31)
FLC	Fuzzy Logic Control (8)
FLN	Functional Link Networks (19)
H	generator inertia constant (158)
IC	Integrated Circuit (19)
I/O	Input-Output (120)
$J_c(k)$	controller cost function at time step k (53)
$J_i(k)$	identifier cost function at time step k (52)
K_A	AVR gains (159)
K_C, K_F	AVR gains (159)
K_{LR}, I_{LR}	AVR gains (159)
K_d	generator damping ratio coefficient (158)
K_s	conventional power system stabilizer gain (105)
L	load in admittance (168)
MLP	Multi-Layer Perceptron (19)
MMI	Man-Machine-Interface (120)
NAPSS	Neural Adaptive Power System Stabilizer (13)
NARMA	Non-linear Auto Regressive Moving Average (47)
NN	Neural Networks (10)
PA	Pole Assignment (9)
PC	Personal Computer (115)
PDP	Parallel Distributed Processing (10)
PHSC	Programmable High Speed Controller (15)
PLC	Programmable Logic Controller (15)
PSS	Power System Stabilizer (5)
PS	Pole Shifting (9)
P_e	generator active power (61)
Q	generator reactive power (168)
RAM	Random Access Memory (122)
RBF	Radial Basis Function (19)
R_C, X_C	voltage transducer compensation constants (159)
SISO	Single-Input Single-Output (21)

SUN	SUN Micro-Systems Inc. (126)
$T_1, \ldots, T_5$	conventional power system stabilizer time constants (105)
T_6	conventional power system stabilizer time constant (160)
T_A, T_R, T_F	AVR time constants (159)
T_B, T_{B1}	AVR time constants (159)
TCR	Time Constant Regulator (115)
T_C, T_{C1}	AVR time constants (159)
TMS320C30	Texas Instruments DSP board (117)
T'_{do}	generator d-axis transient time constant (167)
T''_{do}	generator d-axis sub-transient time constant (167)
T_e	generator electrical torque output (158)
T_g	governor time constant (159)
Θ	bias vector (22)
T_m	generator mechanical torque input (158)
T''_{qo}	generator q-axis sub-transient time constant (167)
$U_{PSS}(s)$	Laplace transform of power system stabilizer output (105)
U_{PSS}	power system stabilizer output (120)
V_{AMAX}	AVR command signal upper limit (159)
V_{AMIN}	AVR command signal lower limit (159)
V_{IMAX}	AVR input upper limit (159)
V_{IMIN}	AVR input lower limit (159)
VLSI	Very Large Scale Integrated (19)
V_{OEL}	AVR over-excitation limit (159)
V_{RMAX}	voltage regulator upper limit (159)
V_{RMIN}	voltage regulator lower limit (159)
V_{STMAX}	PSS output upper limit (159)
V_{STMIN}	PSS output lower limit (159)
V_{UEL}	AVR under-excitation limit (159)
V_b	infinite-bus voltage (72)
V_{ref}	AVR reference voltage (120)
V_t	terminal voltage (72)
$W_c(k)$	matrix of controller weights at time step k (54)
$W_i(k)$	matrix of identifier weights at time step k (53)

W	weight matrix of elements w_{ji} (22)
a, b	governor gain constant (159)
α	momentum factor (32)
δ_{pj}	local gradient of neuron j for pattern p (31)
δ	Dirac delta function (23)
δ	generator power angle (158)
d_{pj}	desired output of neuron j for pattern p (31)
$e_c(k)$	controller error at time step k (53)
e''_d	generator d-axis sub-transient voltage (167)
e_d	generator d-axis voltage (158)
e_f	generator field voltage (158)
$e_i(k)$	identifier error at time step k (52)
e'_q	generator q-axis transient voltage (167)
e''_q	generator q-axis sub-transient voltage (167)
e_q	generator q-axis voltage (158)
η_c	controller learning rate (54)
η_i	identifier learning rate (53)
η	learning rate (31)
$\{f'_i\}, \{g'_i\}$	digital conventional power system stabilizer coefficients (127)
g	governor output (158)
h	tuning parameter (53)
i_d	generator d-axis current (158)
i_f	generator field current (158)
i_{kd}	generator d-axis damper winding current (158)
i_{kq}	generator q-axis damper winding current (158)
i_q	generator q-axis current (158)
λ_d	generator d-axis flux linkage (158)
λ_f	generator field flux linkage (158)
λ_{kd}	generator d-axis damper winding flux linkage (158)
λ_{kq}	generator q-axis damper winding flux linkage (158)
λ_q	generator q-axis flux linkage (158)
ω_0	speed base value (158)
$\dot{\omega}$	generator acceleration (158)

w_{ref}	governor speed reference (72)
w	generator speed (72)
pf	power factor (61)
pu	per unit (61)
r_a	generator armature resistance (158)
r_f	generator field resistance (158)
r_{kd}	generator d-axis damper winding resistance (158)
r_{kq}	generator q-axis damper winding resistance (158)
r_t	transmission line resistance (159)
s	Laplace variable (23)
$tanh$	Hyperbolic tangent (25)
τ	sampling period (127)
θ_i	bias of neuron i (22)
$u(k)$	power system stabilizer output at time step k (60)
$w_{ji}(n)$	weight connecting output of neuron j to input of neuron i at iteration n (22)
x'_d	generator d-axis transient reactance (167)
x''_d	generator d-axis sub-transient reactance (167)
x_d	generator d-axis reactance (159)
x_f	generator field reactance (159)
x_{kd}	generator d-axis damper winding reactance (159)
x_{kq}	generator q-axis damper winding reactance (159)
x_{md}	generator d-axis mutual reactance (159)
x_{kq}	generator q-axis mutual reactance (159)
x''_q	generator q-axis sub-transient reactance (167)
x_q	generator q-axis reactance (159)
x_t	transmission line reactance (159)
y_{pj}	actual output of neuron j for pattern p (31)
ζ	parameters of Hopfield network (32)
z^{-1}	backward discrete-time shift operator (127)

Chapter 1

Introduction

1.1 Power System Stability

Electric power systems are highly complicated systems that contain non-linear and time-varying elements. Their dynamics cover a wide spectrum of phenomena, which are electrical, electro-mechanical and thermal in nature. Since interconnected power systems can encompass entire countries and continents, they can involve a large number of interacting systems with an immense array of variables [1], [2], [3].

The highly interconnected nature of power systems makes their operation and control a complex process. Disturbances in some elements may affect the whole system operation and stability causing poor power quality or even the interruption of power supply [4], [5].

The problem of power system instability first arose when generating units were tied together to improve power system reliability and to reduce the cost of generation [6]. It was noticed that the system damping was insufficient. One of the first approaches to overcome this problem was to introduce damper windings in the synchronous generator [6].

Extensive research has been conducted to overcome power system stability problems. For analytical studies, researchers have classified the power system stability into three categories [7], [8], [9], [10]:

- *Steady-state stability*

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

- *Dynamic stability*

  Dynamic stability is the stability of the power system under small and sudden disturbances. These type of disturbances can lead to long term sustained oscillations [8].

- *Transient stability*

  Transient stability refers to the ability of the power system to regain 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.

A small signal perturbation model around an equilibrium point can be considered for dynamic stability studies and the system can be described by linear differential equations. However, for transient stability analysis and control design, the power system must be described by non-linear differential equations.

Although there are several sources of positive damping [8] in a power system, there are also sources of negative damping, notably voltage-regulating and speed-governing systems. Furthermore, although ordinarily the inherent positive damping predominates, in some circumstances the net damping can become negative. With net negative damping, angular swing of the machine, instead of declining, increases either until equilibrium amplitude is reached or synchronism is lost.

As power system stability is ultimately concerned with the quality of electricity supply, it is one of the main research topics in power system studies. There are three means of improving power system stability:

- Generator excitation control [11], [12], [13], [14], [15], [16], [17].
- Generator input power control [18], [19], [20].
- System operating condition and configuration control [21], [22], [23], [24], [25].

For a particular problem, one or more of the above methods can be used. However, excitation control is usually preferred for the following reasons:

- the electrical system has much smaller time constants than the mechanical system;
- an electrical control system is more economical and easier to implement than a mechanical control system;
- because of small loop time constant, an electrical control system is effectively a continuously acting system. Consequently, it gives smooth system response.

1.2 Excitation Control

Excitation controllers have been widely used in power systems since the early 1960's [26]. The purpose of using excitation control is to achieve an acceptable voltage profile at the consumer terminal and to effectively control the reactive power flow in the network. It is generally recognized that high gain, short time constant and high ceiling voltage excitation usually increases both the steady-state and transient stability limits of the system [27], [28]. Although it is also found that this high performance excitation sometimes provides negative damping, it does not seem to have caused any serious problem in its early applications [29].

As the high performance excitation systems became a large percentage in the generating capacity, it became apparent that their actions had a detrimental impact upon the dynamic stability of the power systems. Low frequency oscillations often persisted for long periods of time and in some cases presented limitations on power transfer capability. It has been found that inappropriately chosen controller parameters greatly decrease the system damping and even make it negative at times [19], [26]. A significant amount of research has been conducted on the development of compensating control to provide the required system stability and various methods have been proposed. Generally these methods can be divided into two areas:

- Design new excitation controllers based on modern control theory to replace old ones [30], [31], [32], [33], [34].

- Improve the performance of the presently used excitation controllers by introducing a supplementary control signal [35], [36].

Some examples of the first area are the utilization of optimal control theory, sub-optimal control, bang-bang control and adaptive control.

The typical method in the second area is to use a power system stabilizer (PSS) to extend stability limits by modulating generator excitation to provide damping to the oscillations of synchronous machine rotors relative to one another.

1.3 Power System Stabilizers

As mentioned in Section 1.2, the basic function of a PSS is to modulate the generator excitation to damp out the oscillations of synchronous generator rotors relative to one another. Oscillations of concern typically occur in the frequency range of approximately 0.2 to 2.5 Hz [37]. Insufficient damping of these oscillations may limit the ability to transmit power. The PSS input is one of the following signals or a combination of them:

- Shaft speed deviation
- Bus frequency
- Electric power deviation or accelerating power

The PSS must operate through the "plant" which consists of the generator, the excitation system, and the power system. The basic characteristics of this plant which are significant to stabilizer applications are as follows:

- phase characteristics of the plant are nearly identical to the phase characteristics of the closed loop voltage regulator;

- gain of the plant increases with the generator load;
- gain of the plant increases as the AC system becomes stronger. This effect is simplified with high gain voltage regulators;
- gain of the plant at the oscillation frequencies of concern is proportional to the regulator gain and inversely proportional to the main generator open-circuit time constant and the oscillation frequency;
- phase lag of the plant increases as the AC system becomes stronger. This has the greatest influence with high gain exciters, since the voltage regulator loop crossover frequency approaches that of the oscillation of concern.

1.4 Different Types of Stabilizers

1.4.1 Conventional Power System Stabilizer

The most commonly used PSS, referred to as the conventional PSS (CPSS) is based on the linear model of the power system at some operating point [9], [37]. Usually the operating condition where the control is needed most is chosen [37]. The classical control theory, based on transfer functions, was employed as the design tool for the CPSS. There have been decades of theoretical studies and field experiments. This type of PSS is widely used in power systems and has made a great contribution in enhancing power system dynamic stability [37]. Since the CPSS is designed based on the linear model of a fixed configuration of the power system for a specific operating point, it works well for the configuration and operating condition for which it was tuned.

However, the CPSS performance deteriorates as the system operating conditions and configuration change. In addition, the highly non-linear power system with saturating elements and stochastic nature, makes the control task of the CPSS even more difficult. Therefore, the CPSS faces a problem in the following areas:

- accuracy of the linear model of the power system;
- accuracy of the parameters for that model;
- effective tuning of the model;
- interaction between the various machines;
- tracking of the system non-linearity and operating condition.

Extensive research has been carried out to solve these problems. Different CPSS transfer functions associated with different systems have been proposed [27], [35], [37]. Various tuning techniques have been introduced to effectively tune CPSS parameters [38], [39], [40]. Effective placement and mutual cooperation between the PSSs in multi-machine systems are also presented [41], [42]. To solve the parameter tracking problem, variable structure control theory was introduced to design the CPSS [43]. All this research has resulted in great progress in understanding the operation of the PSS and effectively applying PSS in the power systems. However, it cannot change the basic fact, namely the conventional PSS is a fixed-parameter controller designed for a specific operating point which generally cannot maintain the same quality of performance at other operating points [44]. It is for this reason that adaptive control, "control" that adapts to changing system characteristics, has so much potential to improve power system performance.

1.4.2 Fuzzy Logic Based Power System Stabilizer

One of the modern methods which has recently been used is Fuzzy Logic Control (FLC). Fuzzy control systems are rule-based systems in which a set of fuzzy rules represent a control decision mechanism. FLC based controllers have a number of advantages:

- *Model-free based algorithm*

  This is a property of a larger group of modern control techniques called artificial intelligence (AI) based controllers. Unlike other classical control techniques, AI based methods (including Neural Networks and Fuzzy Logic) do not require the exact mathematical model of the system.

- *Knowledge based algorithm*

  Fuzzy logic control emulates the strategy of a human operator controlling the process.

- *Small development time*

  Since FLC is a simple algorithm, development time is relatively small.

Research on FLC based PSSs is reported in [45], [46], [47]. The FLC based PSS, however, suffers from two important drawbacks; the parameter tuning and lack of adaptation. The latter is of great importance, since adaptation ability is one of the most important features that a PSS should have.

1.4.3 Adaptive Power System Stabilizer

Adaptive control theory provides a possible way to solve the problems mentioned for the CPSS. At each sampling instance, the input and output of the generating unit are sampled, and a mathematical model is obtained by some on-line identification algorithm to represent the dynamic behavior of the system at that instant of time. It is expected that the mathematical model obtained at each sampling period can track changes in the system.

Following the identification of the model, the required control signal for the generating unit is calculated based on the identified model. There are various control strategies; among them are pole assignment (PA) and pole shifting (PS) methods. These control strategies are generally developed by assuming that the identified model is a very close approximation to the generating unit [48], [49], [50]. However, since the power system is a high-order non-linear continuous system, it is hard for the low-order discrete identified model to precisely describe the dynamic behavior of the power system. Consequently, a high-order model is used to represent the power system, which consumes a significant amount of computation time. This in turn limits the control effect, as the system is unable to act at higher sampling rates. This becomes more significant when the oscillation frequency is relatively high. For this type of controller, there is a compromise between the order of the discrete model and the computation time for parameter identification and optimization.

1.4.4 Neural Network Based Power System Stabilizer

1.4.4.1 Neural Networks

Neural Networks (NN) attempt to achieve good performance via dense interconnection of simple computational elements. Their structure is based on the present understanding of biological nervous systems.

In recent years, interest in studying the mechanism and structure of the brain has been increasing. Based on this biological background, recent work has led to the development of new computational models for solving problems such as pattern recognition, fast information processing and adaptation.

In the early 1940s, pioneers of this field studied the potential and capabilities of the adaptation laws involved in neural systems [51], [52]. In 1950s and 1960s, the Perceptron architecture which has subsequently received much attention was developed and its properties and limitations were analyzed [53], [54]. In 1970s, and 1980s, the work reported in [55] and [56] and the parallel distributed processing (PDP) group [57] provided a strong impetus to the area and was the catalyst for much of the subsequent research in this field. Since then, much research on neural networks has been done and today there are several well-defined architectures to apply to a variety of problems.

Neural networks enjoy a variety of advantages:

- *Capability to synthesize complex mappings*

  Neural networks can synthesize complex and transparent mappings which may be very difficult or even impossible to be expressed in mathematical form. Since a neural network is trained by input-output data, a properly trained neural network can perform highly non-linear mappings [57].

- *High speed*

  Due to the parallel mechanism, the NN has the potential to solve the mapping problem much faster than conventional methods and other artificial intelligence methods, such as expert systems.

- *Robustness and fault tolerance*

  Neural networks are robust. Even if the input data is not complete or has some noise, the NN can still produce good results [58].

- *Adaptation ability*

  Neural networks can be trained on-line by using their error performance. This allows the NN to adjust to a new environment easily.

- *Capacity for generalization*

  Neural networks are able to respond properly to the inputs they haven't come across in training. If neural networks are trained properly they are able to generalize the input space.

1.4.4.2 Neural Network Applications in Power Systems

Since the publication of the first paper on the application of NNs in power engineering in February 1989 [59], many papers have been published in this area. Neural networks have been applied in the following fields of power engineering [60]:

- Load forecasting

- Harmonics Prediction
- Machine or plant control
- Generation Expansion
- Capacitor allocation
- Optimal power flow
- Unit commitment
- Economic load dispatch
- State estimation
- Fault detection and diagnosis
- Alarm processing
- Dynamic security assessment
- Contingency analysis
- Machine modeling

1.4.4.3 Why Neural Network Based Power Systems Stabilizer ?

As mentioned before, the CPSS, which is based on deterministic control theory, has some limitations. It has to be designed for a particular operating condition around which a linearized model is obtained. Usually this operating condition is chosen where the control is needed most [37], i.e. the operating condition at which the generating unit is most likely to operate. The high non-linearity, wide range of operating conditions and non-deterministic properties of the actual power systems present problems to the CPSS. In addition, tuning of the CPSS poses another drawback to CPSS. On the other hand, neural networks have the ability to learn non-linear mappings. They also enjoy the very important feature of learning, enabling them to acquire underlying knowledge from input-output data. Using the on-line learning feature of neural networks, it is proposed that the time-varying power plant can be tracked and control signal can be computed accordingly. Because of these inherent features of neural networks, they appear to be able to implement many functions essential to control systems with a higher degree of autonomy [61].

1.5 Dissertation Objective

In this thesis a neural adaptive power system stabilizer (NAPSS) is proposed to replace the conventional PSS. In order to develop NAPSS, the following topics are discussed and studied in this thesis:

- Investigation of the theory of neural networks and discuss the feasibility of each type of neural networks for application in power system control. Select one, among many types of NN which best fits the PSS design.
- Design of an adaptive neural network based PSS. This PSS is directly trained from the output data in each sampling period and utilizes only the input-output data of the generating unit.
- Simulation studies on the performance of NAPSS in a single-machine power system. Comparison of the control capacity of the NAPSS with that of the CPSS.

- Simulation studies on the performance of the NAPSS in a multi-machine power systems. Investigation of the cooperation of the NAPSS with CPSS in damping multi-mode oscillations.

- Hardware implementation and on-line experimental verification of the proposed NAPSS in a laboratory environment.

The aim of this dissertation is to perform studies on the above-mentioned topics and investigate the feasibility of the NAPSS.

1.6 Dissertation Organization

This thesis is composed of 8 chapters divided into three parts:

- **Part I** *Theoretical development:*

  Chapter 2 serves as a brief review of the basic concepts and theories relating to NNs. The classification of neural networks and the structure of single neurons are introduced. Three most popular types of the NNs,the feed-forward multi-layer network, the Hopfield network and the Kohonen self-optimizing feature maps are discussed. Based on a comparison of the features of different types of NNs, the feed-forward multi-layer network is chosen to build the NAPSS.

  The indirect adaptive control method is described in Chapter 3. The controller is trained on-line using back-propagation method. It is designed using feed-forward multi-layer neural networks. Using single-element error vector, the training algorithm is simplified.

- **Part II** *Simulation studies:*

The neuro-identifier design and simulation results are presented in Chapter 4. The plant Model IV along with series-parallel identifier is used to construct the neuro-identifier. The simulation results for the neuro-identifier are also given in this chapter.

The NAPSS structure and its application to a single-machine power system are presented in Chapter 5. The ability of the NAPSS to provide enough damping over a wide range of operating conditions is discussed.

Then, in Chapter 6, detailed simulation studies of the proposed NAPSS under a multi-machine power system environment are given. A five-machine power system which exhibits both local and inter-area modes of oscillations is used to demonstrate the effectiveness of the NAPSS. Self-coordination ability of the NAPSS with CPSS is also shown.

- **Part III** *Experimental tests:*

  Laboratory implementation and experimental tests of the proposed NAPSS on a physical model power system are described in Chapter 7. Real-time tests are performed on this model employing ABB PHSC Programmable Logic Controller (PLC) acting as an Automatic Voltage Regulator (AVR) and a Digital Signal Processor (DSP) acting as stabilizer. For comparison, a digital type CPSS is also implemented in the same environment and tested under the same conditions. Behavior of the NAPSS and CPSS in an actual power system is observed. Details of implementation along with the experimental results are also described in Chapter 7.

Finally, conclusion and comments on further research topics in this area are summarized in Chapter 8.

1.7 Dissertation Contribution

The author believes that the work presented in this dissertation makes original contribution in the following respects:

- development of an adaptive neural network PSS based on on-line trained feed-forward neural networks which combines the inherent advantages of neural networks and good control performance of adaptive control.
- design of the NAPSS without any reference model or teacher signal and without measuring the internal states, only using the output performance of the plant in real-time.
- simplification of on-line training algorithm by making use of single-element error vector.
- verification of the NAPSS in damping multi-mode oscillations in a multi-machine power system.
- laboratory implementation of the proposed NAPSS and experimental tests on a physical model. Although many off-line simulation studies using neural networks are reported in the literature, rarely, if any, experimental real-time tests have been reported.

Part I

Theoretical Development

Chapter 2

Neural Networks

2.1 Introduction

Artificial neural network models have been studied for many years in the hope of achieving human-like performance in the various fields of science. Neural networks appear to be a recent development, although this field was established before the advent of computers. Inspired by neuro-physiologists such as Donald Hebb [52] at McGill University, work in the neural network field began in the 1940s. During the 50s and 60s, researchers integrated biological and physiological insights to produce the first artificial neural network. The early success generated a burst of research activity.

Then, following some failures, neural network research was eclipsed for nearly two decades [62]. In 1983, increased research funding in neural networks opened the flood gates for intense activity in this area. As an example of the pace at which this field has emerged, it is enough to say that the number of identified neural networks grew from 6 in early 1987 to 26 in early 1988. This field is especially exciting today because neural network algorithms and architectures can be implemented in VLSI technology.

Neural networks and control systems community have a long history, which probably began with the Wiener's book *Cybernetics* [63]. The first neuro-controller was developed by Widrow and Smith in 1963 [64]. A simple ADAptive LINear Element (ADALINE) was taught to reproduce a switching curve in order to stabilize and control an inverted pendulum. This ADALINE was one of the first neural networks (the Perceptron being the other [53]). It has a simple architecture that has been used extensively in other neural networks.

During the 70s, Albus proposed the Cerebellar Model Arithmetic Computer (CMAC) as a tabular model of the functioning of the cerebellum and used it to control robotic manipulation. Since the early 80s, the CMAC has been used extensively to model and control highly non-linear processes [65]. During the 80s, many different neural networks and IC architectures were proposed for integrating and extending these algorithms. Reinforcement learning and adaptive critic schemes have been extensively researched [66] and new neural networks such as Multi-Layer Perceptron (MLP) [57], Radial Basis Function (RBF) [67], Functional Link Networks (FLN) [68] and B-Spline [69] have been developed.

2.2 Types of Neural Networks

In recent years, research in the field of neural networks has achieved significant success. Detailed introduction and classification are given in [58], [70], [62].

In [58], the neural networks are classified into different groups according to the data type of inputs and training procedures. These are:

1) Binary inputs

- Supervised training
  - Hopfield net;
  - Hamming net;
- Unsupervised training
  - Carpenter/Grossberg classifier;

2) Continuous-valued inputs

- Supervised training
  - Perceptron;
  - Multi-layer perceptron;
- Unsupervised training
  - Kohonen self-organizing feature maps.

Among these six types, the Hopfield net, the multi-layer perceptron and the Kohonen self-organizing feature maps have been widely used in power engineering.

In this chapter, a brief review is given of the basic concept of neural network, the most popular types of neural networks and their potential application in power system control.

2.3 Basic Elements

The basic processing element of a neural network is called a neuron by analogy with neurophysiology, but other names such as Perceptron [53] or ADALINE [71] are also used. Fig. 2.1 shows a standard and unifying model of a neuron. It has three components:

- a weighted summer;
- a linear dynamic SISO system;
- a non-dynamic non-linear function

Each of these components is considered in turn below.

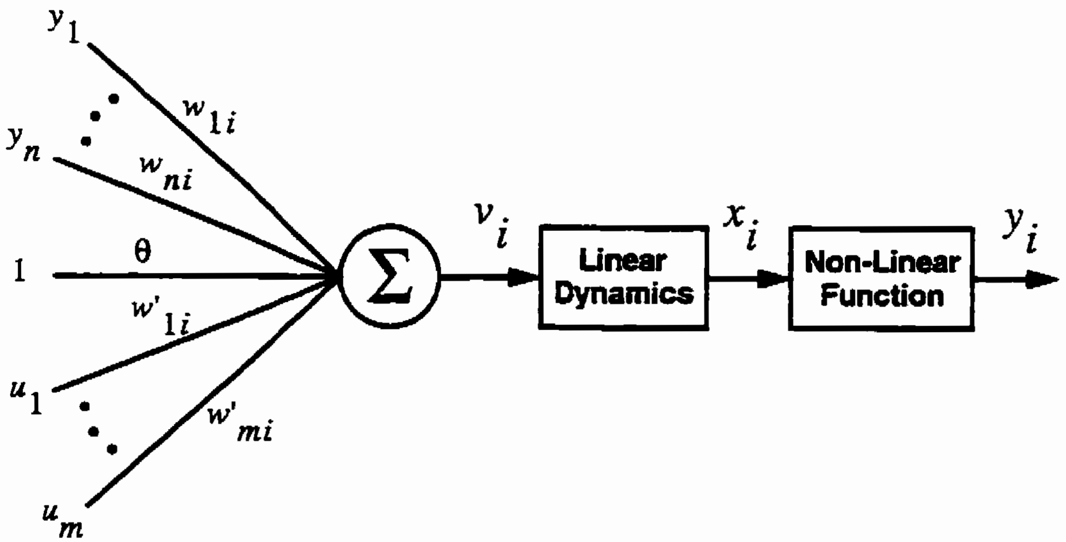


Figure 2.1: Basic model of a neuron.

2.3.1 Weighted Summer

The weighted summer is described by:

$$v_i(t) = \sum_{j=1}^{n} w_{ji} y_j(t) + \sum_{k=1}^{m} w'_{ki} u_k(t) + \theta_i \tag{2.1}$$

giving a weighted sum v_i in terms of the outputs of all elements y_i, external inputs u_k and corresponding weights w_{ji} and w'_{ki} together with the constants θ_i which is called a bias. A number n of these weighted summers can be conveniently expressed in vector-matrix notation.

Stacking n weighted sums v_i into a column vector v, the n outputs y_j into a vector y and m inputs u_k into a vector u and the n constants θ_i into a vector Θ, (2.1) may be written in vector matrix form as:

$$v(t) = Wy(t) + W'u(t) + \Theta \tag{2.2}$$

where the ji-th element of the nxn matrix W is w_{ji} and the ki-th element of the nxm matrix W' is w'_{ki}.

2.3.2 Linear Dynamic System

The linear dynamic SISO system has input v_i and output x_i. In transfer function form it is described by:

$$\bar{x}_i(s) = H(s)\bar{v}_i(s) \tag{2.3}$$

where the bar denotes Laplace transformation. In the time domain, (2.3) becomes:

$$x_i(t) = \int_{-\infty}^{t} h(t-t')v_i(t')dt' \tag{2.4}$$

where $H(s)$ and $h(t)$ form a Laplace transformation pair. Five common choices of $H(s)$ are:

$$\begin{aligned}
H(s) &= 1, && (2.5)\\
H(s) &= \frac{1}{s}, && (2.6)\\
H(s) &= \frac{1}{1+sT}, && (2.7)\\
H(s) &= \frac{1}{\alpha_0 s+\alpha_1}, && (2.8)\\
H(s) &= e^{-sT} && (2.9)
\end{aligned}$$

corresponding to:

$$\begin{aligned}
h(t) &= \delta(t), && (2.10)\\
h(t) &= \begin{cases} 0, & t<0, \\ 1, & t\geq 0, \end{cases} && (2.11)\\
h(t) &= \frac{1}{T}e^{-\frac{1}{T}}, && (2.12)\\
h(t) &= \frac{1}{\alpha_0}e^{-\frac{\alpha_1}{\alpha_0}t}, && (2.13)\\
h(t) &= \delta(t-T) && (2.14)
\end{aligned}$$

where δ is the Dirac delta function. In the time domain, the corresponding input-output relations are:

$$x_i(t) = v_i(t), \tag{2.15}$$

$$\begin{aligned}
\dot{x}_i(t) &= v_i(t), && (2.16)\\
T\dot{x}_i(t) + x_i(t) &= v_i(t), && (2.17)\\
\alpha_0\dot{x}_i(t) + \alpha_1 x_i(t) &= v_i(t), && (2.18)\\
x_i(t) &= v_i(t-T) && (2.19)
\end{aligned}$$

The first, second and third versions are clearly special cases of the fourth.

Discrete-time dynamic systems are also used. For example:

$$\alpha_0 x_i(t+1) + \alpha_1 x_i(t) = v_i(t) \tag{2.20}$$

where t is now an integer time index.

2.3.3 Non-Dynamic Non-Linear Function

The non-dynamic non-linear function $g(.)$ gives the element output y_i in terms of the transfer function output x_i:

$$y_i = g(x_i) \tag{2.21}$$

There are a number of two-fold classifications of these functions:

1) Differentiable/non-differentiable

2) Pulse-like/step-like

3) Positive/zero-mean

Classification 1 distinguishes smooth from sharp functions. Smooth functions are needed for some adaptation algorithms such as back-propagation

[72] (Section 2.5.2), whereas discontinuous (e.g. signum) functions are needed to give a true binary output.

Classification 2 distinguishes functions which only have a significant output value for inputs near zero from functions which change significantly only around zero.

Classification 3 refers to step-like functions. Positive functions change from 0 at $-\infty$ to 1 at ∞; zero-mean changes from -1 at $-\infty$ to 1 at ∞.

Some standard functions are given in Table 2.1. Note that in the table there are strong relations between the given functions. The sigmoid and $tanh$ functions are similar; sigmoid ranges from 0 to 1 while $tanh$ ranges from -1 to 1. Secondly, the threshold functions correspond to the high gain limits of the sigmoid and $tanh$ functions.

2.4 Connections

The neurons by themselves are not very powerful in terms of computation or representation, but their interconnection allows one to encode relations between the variables and gives different powerful processing capabilities. The three components of the neuron discussed in Section 2.3 can be combined in various ways. For example, if the neurons are all non-dynamic ($H(s) = 1$) then an assembly of neurons can be written as the set of algebraic equations obtained by combining (2.2) and (2.3), and (2.21):

$$x(t) = Wy(t) + W'u(t) + \Theta, \tag{2.22}$$

$$y(t) = g(x(t)), \tag{2.23}$$

where x is a vector of n x_i elements and $g(x)$ is a vector whose components are $g(x_i)$. If, on the other hand, each neuron has first order low-pass dynamics:

$$H(s) = \frac{1}{sT+1} \tag{2.24}$$

Table 2.1: Non-linear functions, $g(x)$.

Name	Formula	Characteristics
Threshold	1 if $x > 0$, 0 otherwise	Non-differentiable, Step-like, Positive
Threshold	$+1$ if $x > 0$, -1 otherwise	Non-differentiable, Step-like, Zero-mean
Sigmoid	$\frac{1}{1+e^{-x}}$	Differentiable, Step-like, Positive
Hyperbolic tangent	$tanh(x) = \frac{e^{x_i}-e^{-x_i}}{e^{x_i}+e^{-x_i}}$	Differentiable, Step-like, Zero-mean
Gausian	$e^{(\frac{-x^2}{\sigma^2})}$	Differentiable, Pulse-like,

then an assembly of neurons can be written as the set of differential equations:

$$
\begin{aligned}
T\dot{x}_i(t) + x_i(t) &= Wy(t) + W'u(t) + \Theta, && (2.25)\\
y(t) &= g(x(t)) && (2.26)
\end{aligned}
$$

Clearly, the solutions of (2.22) and (2.23) form possible steady-state solutions of (2.25) and (2.26).

Discrete-time versions of (2.25) and (2.26) are:

$$
\begin{aligned}
Tx(t+1) + (1-T)x(t) &= Wy(t) + W'u(t) + \Theta, && (2.27)\\
y(t) &= g(x(t)) && (2.28)
\end{aligned}
$$

where t is the integer time index.

The behavior of such a network clearly depends on the interconnection matrix W and on the form of $H(s)$. The variations of W and $H(s)$ lead to different types of neural networks. Three most popular types of neural networks are discussed in the following sections.

2.5 Multi-Layer Feed-Forward Network

The multi-layer feed-forward network is also called the Multi-Layer Perceptron (MLP) [58], [62] or madaline [70]. It is widely applied in power engineering. Almost 60% of neural networks applications in power engineering are based on multi-layer networks, and almost 90% of the neural networks applications in control systems employ this type of neural network.

2.5.1 Configuration

Generally the connection of several layers gives the possibility of more complex non-linear mapping between the inputs and the outputs. This capability can be used to implement classifier or to represent complex non-linear relations among the variables.

Such networks are typically non-dynamic; that is $H(s) = 1$ in (2.3). The connection matrix W is such that the outputs are partitioned into layers so that a neuron in one layer receives inputs only from neurons in the previous layer (or, in the case of first layer, from the network input layer). The elements of the connection weight matrix W are derived from the training process. There is no feedback in such networks.

A four layer network (an input layer, two hidden layers and one output layer) is shown in Fig. 2.2. Neurons in the first hidden layer receive the inputs from the external inputs and send the outputs to the second hidden layer. The neurons in the second hidden layer receive the outputs of the first hidden layer as their inputs, and send their outputs to the output layer. Neurons in the output layer get the outputs from the second hidden layer as the inputs, and the outputs of the output layer are the outputs of the neural network.

Each neuron i gets the weighted sum of the outputs of all the neurons j in the previous layer that connect with neuron i through weight w_{ji}, which is given as:

$$x_i = v_i = \sum_j w_{ji} y_j + \theta_i \tag{2.29}$$

If neuron i is in the first layer, the weighted sum is over all of the external

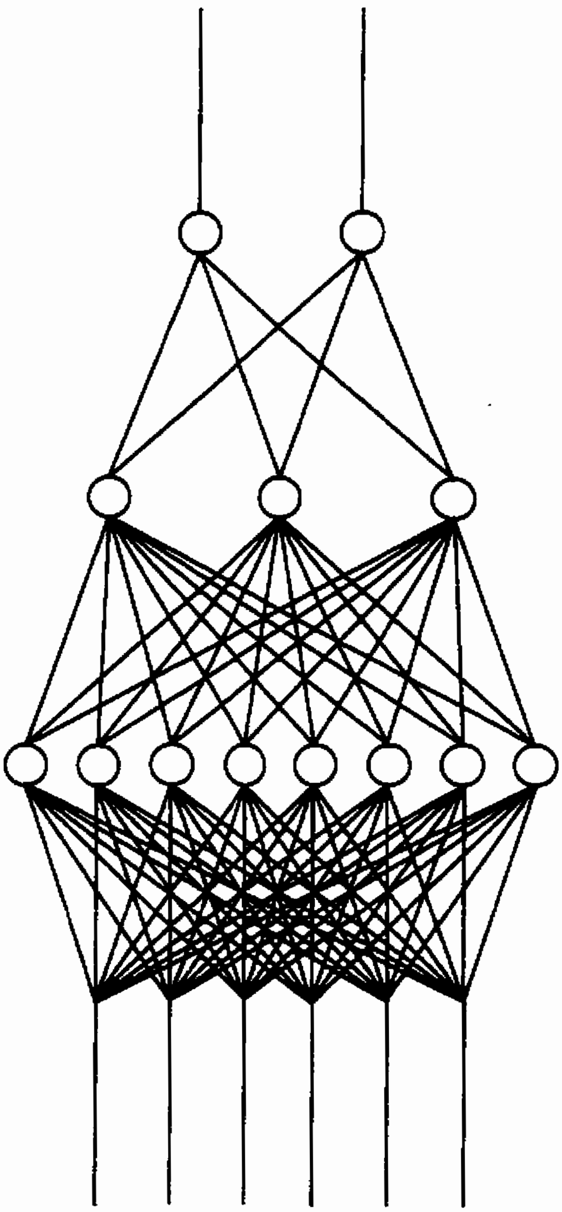


Figure 2.2: Multi-layer network with two hidden layers.

inputs k that connect with neuron i through weight w'_{ki} as shown below:

$$x_i = v_i = \sum_k w'_{ki} u_k + \theta_i \tag{2.30}$$

where θ_i is a bias of the neuron i.

This weighted sum is altered by a non-linear function to establish the output. Since the back-propagation training method, which will be discussed later in this section, requires differentiable non-linear functions, and the nature of control system requires a zero-mean control signal, the most appropriate non-linear function is a hyperbolic tangent-like function, which is given as:

$$y_i = \frac{e^{x_i} - e^{-x_i}}{e^{x_i} + e^{-x_i}} r \tag{2.31}$$

where r is the maximum absolute value of the neuron output.

2.5.2 Back-Propagation Training Method

Connecting weights between the neurons must be determined before the neural network can be used in the application. The process of determining the weights is called the training or learning process. The multi-layer network employs the back-propagation method which was developed in [73] for its training.

The learning procedure proposed here involves the presentation of a set of pairs of input and output patterns. For each input and output pattern p, the system first uses the input vector to produce its own output vector and then compares this with the desired output, or target vector. If there is no difference, no learning takes place. If a difference exists, the weights are adjusted to eliminate the total squared error, E_p, which is the sum of the squared differences between the set of desired outputs and the set of actual outputs of the neural network:

$$E_p = \frac{1}{2}\sum_j (d_{pj} - y_{pj})^2 \tag{2.32}$$

where d_{pj} is the j-th desired output of pattern p, and y_{pj} is the j-th actual output of pattern p.

The weights w_{ij} can be adjusted to minimize E_p for the set of training patterns by a gradient descent method:

$$w_{ij}(n+1) = w_{ij}(n) + \Delta w_{ij}(n) \tag{2.33}$$

where:

$$\Delta w_{ij}(n) = \eta \delta_{pj} y_{pi} \tag{2.34}$$

where n is the iteration number, and η is the learning rate. If the neuron j is in the output layer:

$$\delta_{pj} = \frac{dy_{pj}}{dx_{pj}}(d_{pj} - y_{pj}) \tag{2.35}$$

If the neuron j is not in the output layer:

$$\delta_{pj} = \frac{dy_{pj}}{dx_{pj}}\sum_k \delta_{pk} w_{kj} \tag{2.36}$$

where δ_{pk} is from the neurons in the layer following the layer where neuron j is located.

Better convergence can be achieved if a momentum term is added to (2.34) as:

$$\Delta w_{ij}(n) = \eta\delta_{pj}y_{pi} + \alpha\Delta w_{ji}(n-1) \tag{2.37}$$

where α is the momentum factor. In the above equations, the learning rate, η, and the momentum factor, α, are between 0.0 and 1.0 to be determined by experience. Some good discussion on selecting η and α is given in [74].

The configuration of the multi-layer network has to be determined by experience since there are no definite rules to select the number of hidden layers and the number of neurons in each hidden layer.

2.6 Hopfield Net

The introduction of feedback produces a dynamic network with several stable points. The general equation can be expressed as:

$$\dot{x}(t) = F(x(t), u(t), \zeta), \tag{2.38}$$

$$y(t) = G(x(t), \zeta) \tag{2.39}$$

Here, x represents the state, u the external inputs, and ζ the parameters of the network. F is a function that represents the structure of the network and G is a function which represents the relation between the state variables and the outputs.

Originally, feedback (recurrent) networks were introduced in the context of associative or content addressable memory problems for pattern recognition. The uncorrupted pattern is used as a stable equilibrium point and its noisy versions should lie in its basin of attraction. In this way, a dynamical system associated with a set of pattern is created. If the whole working space is correctly partitioned by such a content-addressable memory, then any initial condition should have a steady-state solution corresponding to the uncorrupted pattern. The dynamics of such a classifier serve as a filter.

The best-known example of a content-addressable memory is the Hopfield net [75]. The structure of the Hopfield net with n neurons is shown in Fig. 2.3. There are two models of the Hopfield net, the discrete model and the continuous model.

2.6.1 Discrete Model of Hopfield Net

The discrete model assumes the step-like non-linearity:

$$g(x_i(t)) = \begin{cases} 0, & x_i(t) < 0, \\ 1, & x_i(t) > 0, \end{cases} \tag{2.40}$$

with:

$$x_i(t) = \sum_{j=1}^{n} w_{ji} y_j(t) + \theta_i \tag{2.41}$$

and works in the asynchronous mode, i.e. only one neuron output is calculated at a time, leaving the others unchanged. The active neuron, p, is chosen randomly. The system evolves with weights w_{ji} established earlier, which will be discussed below, and held fixed during output calculation. The update rule is as follows:

$$y_i(t+1) = \begin{cases} g(x_i(t)), & \text{if } i = p,\ x_i(t) \neq 0 \\ y_i(t), & \text{otherwise} \end{cases} \tag{2.42}$$

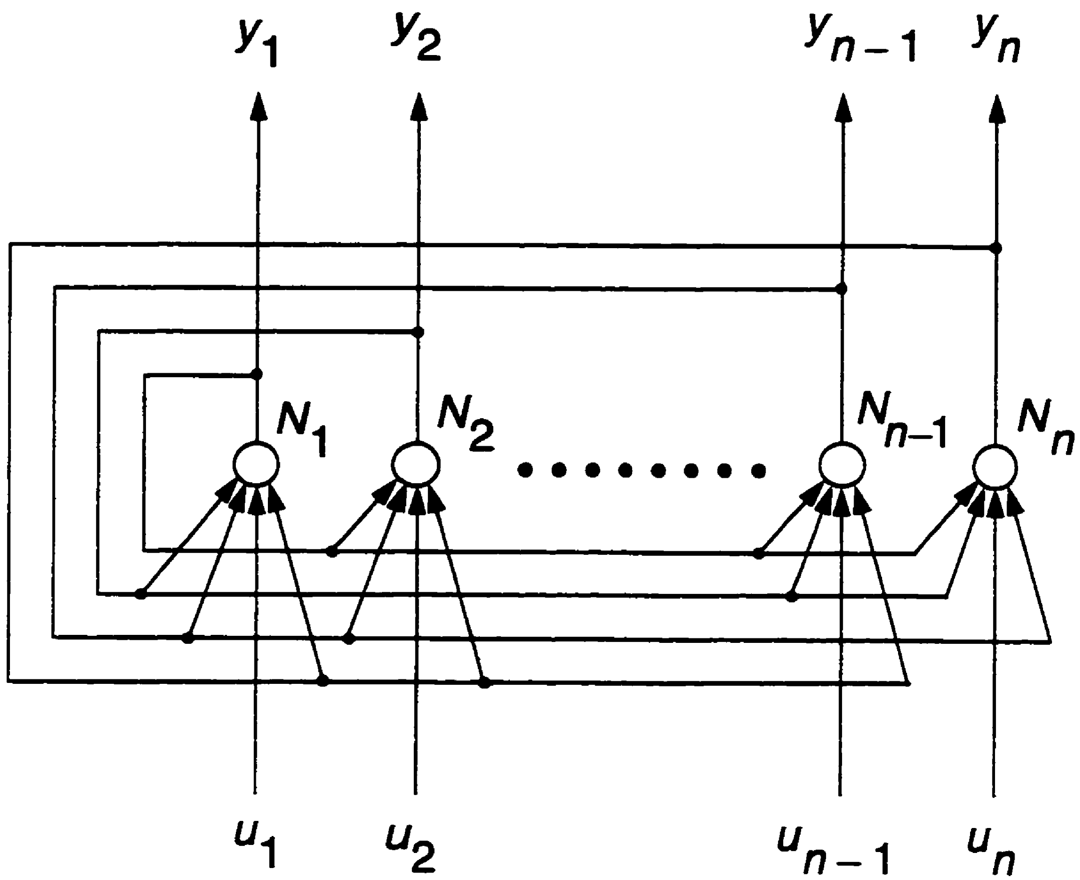


Figure 2.3: The Hopfield net.

If $w_{ii} = 0$ and $w_{ji} = w_{ij}$, then the energy function is given by

$$E(y) = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}y_iy_j + \sum_{i=1}^{n}\theta_i y_i \tag{2.43}$$

or

$$E(y) = -\frac{1}{2}y^T W y + \Theta^T y \tag{2.44}$$

will decrease with every asynchronous change of y_p according to

$$\triangle E = -\triangle y_p[\sum_{j=1}^{n} w_{pj}y_j - \theta_j] \tag{2.45}$$

where

$$\triangle y_p(t) = y_p(t+1) - y_p(t) \tag{2.46}$$

The network will always reach an equilibrium because (2.43) and (2.44) are bounded and (2.45) is non-positive, and the system does not change when $\triangle E = 0$. It will settle after a finite time, since the domain of E is finite.

The Hebbian rule is an attempt to encode P patterns, y_k, $k = 1, \ldots, P$, as equilibrium points of the system represented by (2.40) to (2.46) by choosing

$$w_{ij} = \begin{cases} \sum_{k=1}^{P}(2y_i^k - 1)(2y_j^k - 1), & \text{if } i \neq j, \\ 0, & \text{otherwise} \end{cases} \tag{2.47}$$

and

$$\theta_i = \frac{1}{2}\sum_{j=1}^{n} w_{ij} \tag{2.48}$$

since E is within a multiple and is a constant

$$E \sim -(2y-\vec{1})^T[\sum_{k=1}^{P}(2y^k-\vec{1})^T(2y^k-\vec{1})](2y^k-\vec{1}) \tag{2.49}$$

where $\vec{1}$ is a vector of 1s. If $(2y^k - \vec{1})$ are all orthogonal, E will have a minimum at each y^k, and, hopefully, the dynamics of the system will have a region of attraction about each y^k that associates initial values of y that are near y^k.

2.6.2 Continuous Model of the Hopfield Net

The continuous model is described by:

$$T_i\dot{x}_i = -x_i + \sum_{j=1}^{n} w_{ji}y_j + \theta_i \tag{2.50}$$

$$y_i = g_i(x_i), i = 1,\ldots,n \tag{2.51}$$

where $x_i = x_i(t)$, $y_i = y_i(t)$ and $g(.)$ is sigmoid function. The system described by (2.50) and (2.51) is just a system of ordinary differential equations. Hopfield suggested the Lyapunov function:

$$E = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}y_iy_j + \sum_{i=1}^{n}\rho_i\int_0^{y_i} g_i^{-1}(\xi)d\xi - \sum_{i=1}^{n}\theta_iy_i \tag{2.52}$$

where $\rho_i > 0$ are constants, $g_i(.)$ are monotone increasing functions and $w_{ji} = w_{ij}$ for all i and j. It is straightforward to show that $\dot{E} \leq 0$.

2.7 Kohonen Self-Organizing Feature Maps

It is believed that the placement of neurons in the brain is orderly and reflects some physical characteristic of the external stimulus being sensed. Although much of the low-level organization is generally pre-determined, it is likely that some of the organization at high level is created during learning by algorithms which promote self-organizing. Kohonen self-organizing feature maps [76] are similar to those that occur in the brain.

Kohonen's algorithm creates a vector quantizer by adjusting weights from common input nodes to m output nodes arranged in a two dimensional grid as shown in Fig. 2.4. Output nodes are extensively interconnected with many local connections. Continuous-valued input vectors are presented se-

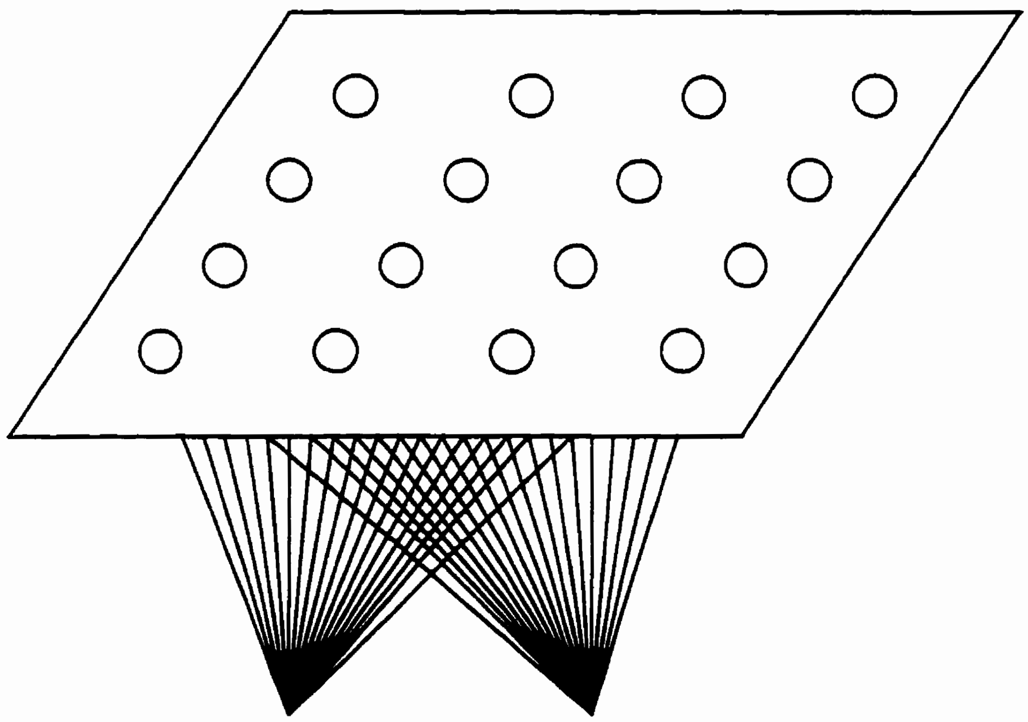


Figure 2.4: Kohonen self-organizing feature map.

quentially in time without specifying the desired output to train the nets. Before the actual training, the weights from n inputs to the m output nodes are initialized to small random values. After the presentation of each new input, the distance d_j between the input and each output node j is computed using:

$$d_j = \sum_{i=1}^{n}(u_i(t) - w_{ij}(t))^2 \tag{2.53}$$

where $u_i(t)$ is the input to node i at time t and $w_{ij}(t)$ is the weight from input node i to output node j at time t. Then the weights for node J which has minimum d_J and the weights for the neighbors of node J are updated. The new weights are:

$$w_{ij}(t+1) = w_{ij}(t) + \eta(t)(u_i(t) - w_{ij}(t)), 1 \leq i \leq n \tag{2.54}$$

where $\eta(t)$ is a gain term $(0 < \eta(t) < 1)$.

2.8 Suitability of Different NNs in Control System Applications

In the previous sections, three most popular types of neural networks have been discussed. Due to the different properties they have, these three types of neural networks can be applied in control systems in different ways.

Hopfield nets are mainly used as associative memories or as classifiers. They can be used to memorize a control function required by a control system. The continuous model is more appropriate in control systems. If the discrete model is to be used, A/D and D/A converters are needed for continuous input and output values. Hopfield nets have two major limitations in application as controllers. The first limitation is that the number of patterns of the control outputs that can be stored and accurately recalled is severely limited. Hopfield showed that the number of classes M must be less than 15% of the number of nodes N [56]. For a fairly complicated system like a generating unit, an extremely large number of nodes in a Hopfield net will be required. One practical method to overcome this limitation is to use a pre-processor to simplify the input and output patterns. The second limitation is that the time of convergence is unpredictable. For a real-time application, the longest running time must be considered.

The properly trained Kohonen self-organizing feature maps will map different inputs to different output nodes and map the inputs with similar features into closer output nodes. Since the training of this type of neural network is unsupervised, the self-organizing feature map cannot be used directly as a controller, but it can be used as a pre-processor for other neural network controllers.

The feed-forward multi-layer network is the most commonly used neural network in control systems. Such networks can generate input/output maps which can approximate, under mild assumptions, any static function with any desired accuracy. One may have to use a large number of neurons , but any desired approximation, if it can be accomplished at all, can be accomplished with a multi-layer network with only one hidden layer of neurons or two layers of weights [61].

Compared with other types of neural networks, the feed-forward multi-layer neural network is more appropriate for application in control systems.

In the proposed neural adaptive power system stabilizer (NAPSS) in this dissertation, the multi-layer network with on-line back-propagation learning is employed to build the adaptive neural network controller.

2.9 Summary

The basic concepts and theories of neural networks are introduced in this chapter. According to the data types of inputs, the architecture , and the training procedure, neural networks are classified into different groups. The basic processing element of a neural network, the neuron, has three components, i.e. weighted summer, linear dynamic SISO system, and non-dynamic non-linear function. The three components of the neurons can be combined in various ways, that distinguishes various neural networks from one another.

Three most popular types of neural networks, the feed-forward multi-layer network, the Hopfield net and the Kohonen self-organizing feature map are discussed in this chapter.

The feed-forward multi-layer network is a non-dynamic network. It has an output layer, an input layer and several hidden layers. The information can only be fed forward between layers. There is no feedback information available during the operation. However, the feedback information is available during the training by using the back-propagation training method.

Hopfield net is a dynamic network with feedback. All of the neurons are interconnected into one layer. This net has two models, the discrete model and the continuous model.

The Kohonen self-organizing feature maps are designed to simulate the low-level organization in the brain, and employ the unsupervised training algorithm. Properly trained Kohonen self-organizing feature maps will map different inputs to different output nodes and map the inputs with similar features into closer output nodes.

Comparing the features of three neural networks, the feed-forward multilayer neural network is more appropriate for application in control systems. Therefore, it is employed to build the neural network controller in this dissertation.

Chapter 3

Indirect Adaptive Control Using Neural Networks

3.1 Introduction

Studies over the past four decades have shown that power system stabilizer (PSS) is a very effective tool to damp out the low frequency oscillations in the power system. Since power systems are highly non-linear dynamic systems, design of a PSS which can maintain the desired performance under different operating conditions is a topic of continuing investigation. Conventional power system stabilizer (CPSS) is designed based on linear control theory [8], [37]. The parameters of the CPSS are usually fixed at a certain set of values which are determined based on a nominal operating condition [42]. Therefore, the fixed parameter CPSS is a compromise between the best settings for light and heavy load conditions. As a result, it is impossible for this type of stabilizers to maintain the best damping performance when there is a drastic change in the system operating conditions, such as that resulting from a three phase to ground fault.

In order to overcome this problem, an adaptive stabilizer should be used in which the parameters are adjusted on-line to automatically track the variations in the operating conditions and system structure. For over 20 years, two distinct approaches have been used to control a plant adaptively [77]. These are *Direct Adaptive Control* and *Indirect Adaptive Control*. In direct control [78], [79], the parameters of the controller are directly adjusted to reduce some norm of the output error. In indirect control [78] , [80], [81], the parameters of the plant are estimated as the elements of a vector $\hat{p}(k)$ at any instant k and the parameters vector $\theta(k)$ of the controller is adapted based on that vector. Even when the plant is assumed to be linear and time-invariant, both direct and indirect adaptive control result in overall non-linear systems [78].

Most adaptive control methods require either a reference model or an extensive identification scheme. Use of the reference model is usually avoided in power systems due to the difficulty involved in choosing a proper model for a complex non-linear plant such as a power system. Identification of the power plant, on the other hand, as studied in classical adaptive control, is also a computationally extensive task which increases the complexity of the controller. Therefore, for the purpose of control of power systems, it is desirable to use a method which is neither model reference based nor computationally extensive.

Neural networks have recently emerged as a successful tool in the fields of pattern classification, modeling and control of dynamical systems [58], [78], [82]. This is mainly due to the computational efficiency of the back-propagation algorithm [72], [73] and the versatility of the three layer feed-forward neural network in approximating an arbitrary static non-linear function [83]. A neural network based controller using an indirect adaptive control method is presented in this chapter. It combines the advantages of neural networks with good performance of the adaptive control. The proposed controller employs the learning ability of neural networks in adaptation process and is trained in each sampling period using the on-line version of the back-propagation algorithm [62], [84]. It consist of two subnetworks. The first one is an adaptive neuro-identifier (ANI) which identifies the power plant in terms of its internal weights and predicts the dynamic characteristics of the plant; and the second one is an adaptive neuro-controller (ANC) which provides the necessary control action to damp out the oscillations of the plant output.

3.2 Adaptive Neuro-Identifier

To design an adaptive controller, a suitable identification algorithm must be chosen. The required control signal is, then, computed based on the identifier parameters. Therefore, the identifier plays an important role in the control algorithm.

Identification of a system has three major steps:

- Selection of a suitable plant model;
- Selection of a proper identification model;
- Adjustment of the parameters of the model so as to minimize a certain cost function.

In the following subsections, first, the issues of selecting the proper plant model and a neural network model for the identification purposes are addressed. In the first step, considering the nature of the dependence of the plant output on the past plant inputs and outputs, a plant model is chosen. In contrast to the static systems which are described by algebraic equations, dynamical systems are governed by differential or difference equations. It is, therefore, important to understand how a dynamic system can be modeled by feed-forward memoryless neural networks.

In the second step, a proper identification model should be selected based on the availability of the plant states. In general, a model is based on the plant states. However, for the cases where only outputs are available, it is possible, under certain assumptions, to predict the output from delayed inputs and outputs [78], [85] using a Multi-Layer Perceptron (MLP). In this dissertation, it is assumed that the states of the plant are not accessible and hence the identifier is based on the inputs and outputs of the plant.

And finally for the third step of identification, i.e., parameter update, the well-known back-propagation algorithm is used in an on-line mode to be suitable for adaptive control methods. To maintain the simplicity of the learning method, only a scalar error, as opposed to a vector of delayed errors, is back-propagated through the neural network.

3.2.1 Plant Models

In this subsection four models of the discrete-time plants are introduced [78], [86], [87]. They can be described by the following equations:

Model I:

$$y(k+1) = \sum_{i=0}^{n-1} a_i y(k-i) + g[u(k), u(k-1), \ldots, u(k-m+1)] \tag{3.1}$$

Model II:

$$y(k+1) = f[y(k), y(k-1), \ldots, y(k-n+1)] + \sum_{i=0}^{m-1} b_i u(k-i) \tag{3.2}$$

Model III:

$$\begin{aligned} y(k+1) = f[y(k), y(k-1), \ldots, y(k-n+1)] + \\ g[u(k), u(k-1), \ldots, u(k-m+1)] \end{aligned} \tag{3.3}$$

Model IV:

$$\begin{aligned} y(k+1) = h[y(k), y(k-1), \ldots, y(k-n+1), \\ u(k), u(k-1), \ldots, u(k-m+1)] \end{aligned} \tag{3.4}$$

where $u(k)$ and $y(k)$ represent the input and the output of the plant at time k respectively. The functions f: $\mathcal{R}^n \rightarrow \mathcal{R}$, g: $\mathcal{R}^m \rightarrow \mathcal{R}$ and h: $\mathcal{R}^{n+m} \rightarrow \mathcal{R}$ are assumed to be differentiable functions of their arguments. These functions along with a_i and b_i parameters are found by the identification process. In all four models, the output at time $k+1$ depends on both past n values of output and past m values of input. Models I and II assume linear dependence on the past values of plant output and input, respectively, while Model III assumes decoupled non-linear dependence on the system input and output. Model IV is the most general and covers the other three models as a special case. Thus, for an unknown system or a system which is neither separable nor linear (in input or output), Model IV is used. Since synchronous generator is neither a separable nor a linear system, Model IV is employed for modeling the plant.

3.2.2 Identification Models

There are two main categories for the identification models in system literature [88]; the State-Output model and the Non-linear Auto Regressive Moving Average (NARMA) model. Their main difference roots in the availability of the states for measurement. The following subsections explain the basic procedure for each model.

The State-Output Model

It is well known in system theory that the state-output model, which relates the past and the present states, can represent a fairly large class of non-linear dynamical systems. The state-output model is given by

$$\begin{aligned} \hat{x}(k) &= \Phi[x(k-1), u(k-1)] \\ \hat{y}(k) &= \Psi[\hat{x}(k), u(k)] \end{aligned} \tag{3.5}$$

where $u(k)$ and $x(k)$ represent the input and the state of the system, $\hat{x}(k)$ is the state of the model, $\hat{y}(k)$ is the output of the model and k is the discretized time. The non-linear functions Φ and Ψ are static and hence can be modeled by feed-forward neural networks. If all of the system states, along with the outputs, are measured, then the problems of building Φ and

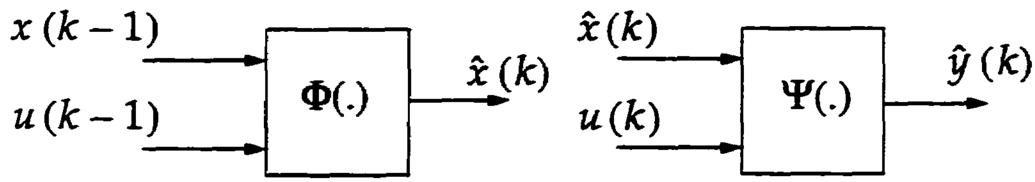


Figure 3.1: The state-output model.

Ψ are decoupled. To train a neural network to approximate Φ, the inputs are $x(k-1)$ and $u(k-1)$ and the output is $\hat{x}(k)$. The inputs to the Ψ block are $\hat{x}(k)$ and $u(k)$ and the output is $\hat{y}(k)$ (Fig. 3.1). Any supervised learning method can be used for training [72], [73].

The NARMA Model

Since all the states are not usually available for measurement, the state-output model, although quite general, is not a good candidate for the identification model. In this case, a NARMA model [78], [88], which relates the output of the plant to the past inputs and outputs of the plant by means of a non-linear function, is preferred. There are two approaches to use a NARMA model; namely the parallel model and the series-parallel model. In the parallel model, the governing equation of the identifier is

$$\begin{aligned}\hat{y}(k+1) = N_p[&\hat{y}(k), \hat{y}(k-1), \ldots, \hat{y}(k-n+1),\\ &u(k), u(k-1), \ldots, u(k-m+1)]\end{aligned} \tag{3.6}$$

where N_p: $\mathcal{R}^{n+m} \to \mathcal{R}$ is a static mapping. This model uses the identifier output for autoregression (Fig. 3.2) which results in slow convergence and sometimes may even lead to instability. Hence, this model is not generally

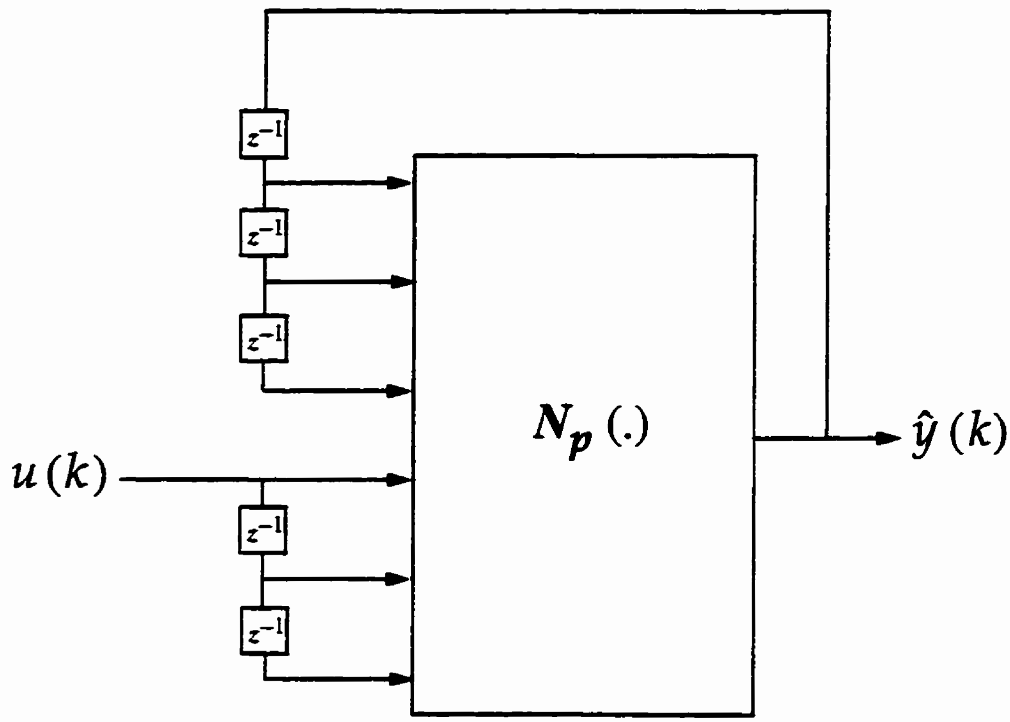


Figure 3.2: The parallel model.

used. Instead, the following model, known as series-parallel model, is used. In this case the identifier equation has the form

$$\hat{y}(k+1) = N_s[y(k), y(k-1), \ldots, y(k-n+1),$$
$$u(k), u(k-1), \ldots, u(k-m+1)] \tag{3.7}$$

In contrast to the parallel model, in the series-parallel model the output of the plant (instead of the model) is fed back to the identifier as shown in Fig. 3.3. In this dissertation, this model is used for the identification of synchronous generator.

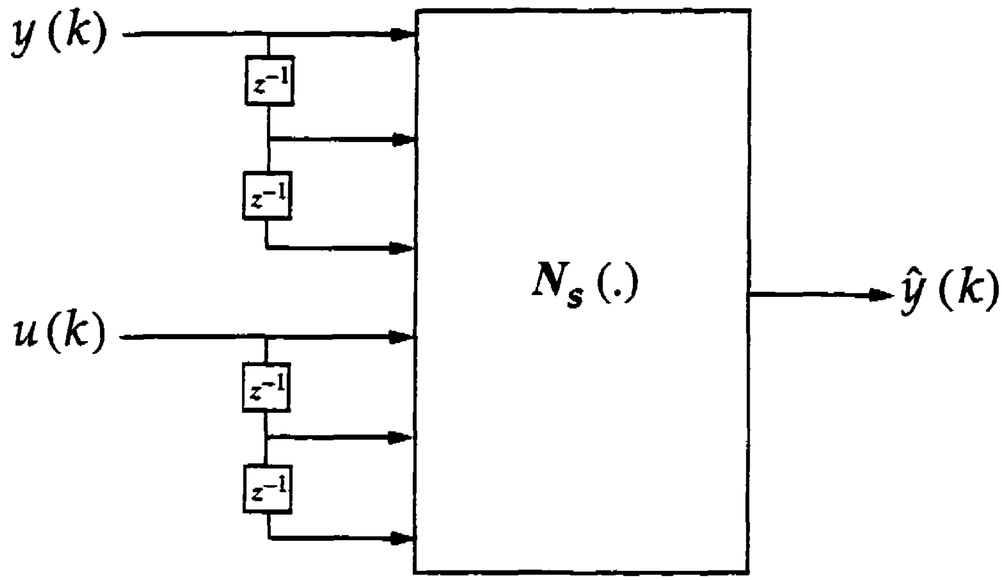


Figure 3.3: The series-parallel model.

3.3 Adaptive Neuro-Controller and Control Algorithm

The structure of the control system is shown in Fig. 3.4. It consists of two subnetworks. The first subnetwork is an adaptive neuro-identifier (ANI) which tracks the dynamic behavior of the plant and identifies the plant in terms of its internal weights, and the second one is an adaptive neuro-controller (ANC) to provide the necessary control action so as to damp out the oscillations of the plant output. This architecture was first introduced in [89]. However, the learning process employed in this dissertation is quite different. Here, a scalar error is used in each sampling period to update the identifier and controller weights continuously. The same architecture is also proposed in [78] and called as *Indirect Adaptive Control.* In that paper, the

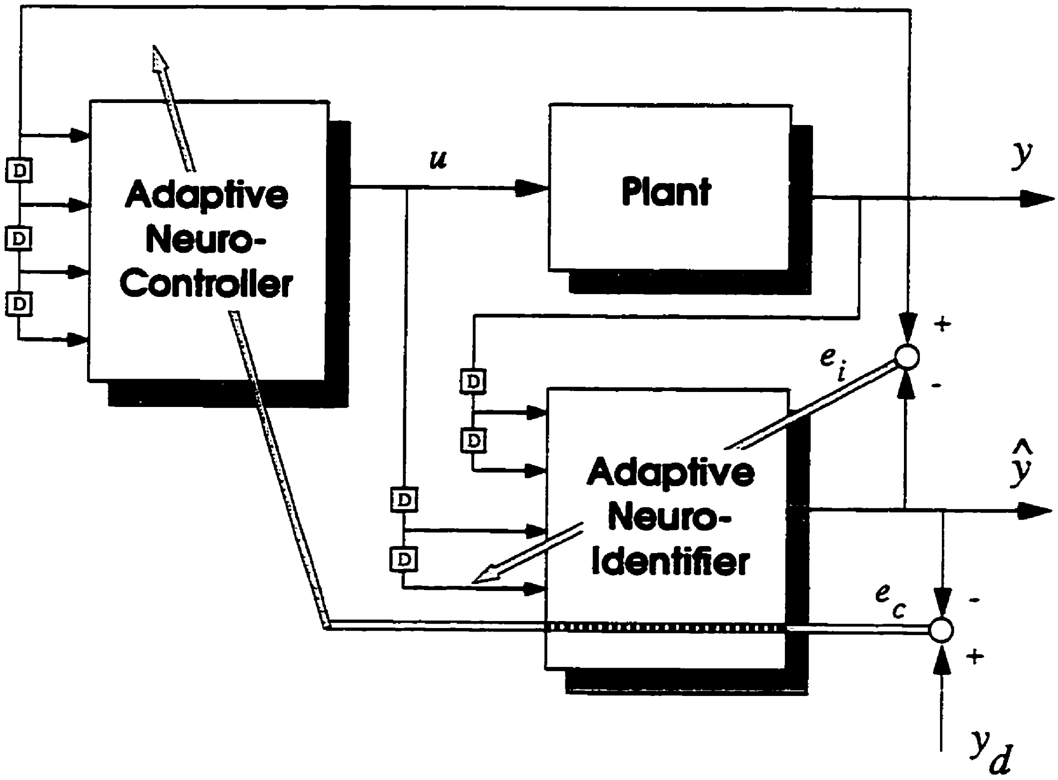


Figure 3.4: Controller structure.

authors suggested the use of a reference model that is avoided here owing to the difficulties in choosing a proper reference model for a complex system such as power system. The next two sections describe the details of each subnetwork.

3.3.1 Adaptive Neuro-Identifier

The input vector to the ANI is :

$$[y(k), y(k-1), \ldots, y(k-m), u(k), u(k-1), \ldots, u(k-n)] \tag{3.8}$$

where $y(k)$ is the plant output and $u(k)$ is the plant input (controller output), both at time step k. The output of the identifier is the predicted plant output, $\hat{y}(k+1)$, at time step $(k+1)$. This is based on considering a series-parallel identifier along with the plant Model IV.

As it is seen, this model is the most general non-linear model which considers coupled non-linear dependence between the plant output at the present time and past values of plant inputs and outputs. The series-parallel identifier, in lieu of parallel identifier, is considered here since the former has faster convergence and is more stable.

The cost function used for the ANI is

$$J_i(k) = \frac{1}{2}e_i(k)^2 = \frac{1}{2}[y(k) - \hat{y}(k)]^2 \tag{3.9}$$

The weights are updated as

$$W_i(k) = W_i(k-1) - \eta_i \nabla_{W_i} J_i(k) \tag{3.10}$$

in which $W_i(k)$ is the matrix of identifier weights at time step k and η_i is the learning rate for the ANI. The gradient $\nabla_{W_i} J_i(k)$ is computed by:

$$\nabla_{W_i} J_i(k) = -[y(k) - \hat{y}(k)] \frac{\partial \hat{y}(k)}{\partial W_i(k)} \tag{3.11}$$

Using (3.10) and (3.11), the cost function $J_i(k)$ is minimized in each sampling period by back-propagating the scalar error $[y(k) - \hat{y}(k)]$.

3.3.2 Adaptive Neuro-Controller

The input vector to the ANC is:

$$[y(k), y(k-1), \ldots, y(k-p)] \tag{3.12}$$

The output of the ANC is the control action, $u(k)$, at time step k. The cost function for the ANC is considered as:

$$\begin{aligned} J_c(k) &= \frac{1}{2}[e_c(k)^2 + hu(k)^2] \\ &= \frac{1}{2}[y_d(k) - \hat{y}(k)]^2 + \frac{h}{2}u(k)^2 \end{aligned} \tag{3.13}$$

where $y_d(k)$ is the desired output at time step k, which is equal to zero in a regulatory setup, and h is a tuning parameter which is used to improve the plant output dynamic characteristics. By taking h greater than zero, a penalty factor is applied to the control action generated by the controller which helps in the tuning of the dynamic trajectory and in optimizing the overshoot and the settling time of the response curve. The weights of the controller, $W_c(k)$, are updated as

$$W_c(k) = W_c(k-1) - \eta_c \nabla_{W_c} J_c(k) \tag{3.14}$$

where η_c is the controller learning rate and the gradient $\nabla_{W_c} J_c(k)$ is defined as

$$\nabla_{W_c} J_c(k) = [\hat{y}(k)\frac{\partial \hat{y}(k)}{\partial u(k)} + hu(k)]\frac{\partial u(k)}{\partial W_c(k)} \tag{3.15}$$

Using (3.14) and (3.15), $J_c(k)$ is minimized in each sampling period.

3.4 Training Process

The success of the control algorithm presented in section 3.3 highly depends on the accuracy of the identifier in tracking the dynamic plant. For this reason, the ANI is initially trained off-line before being hooked up in the final configuration. The training data were collected for operating conditions in the range of 0.1 *pu* to 1.0 *pu* power output and 0.7 *pf* lead to 0.1 *pf* lag. The disturbances used were the voltage reference and input torque reference disturbances as well as three phase to ground fault. The batch mode of back-propagation algorithm with adaptive learning rate was employed. The training was iteratively done until a pre-specified tolerance is met. After the off-line training stage, the ANI is hooked up in the system. Further training of the ANI and ANC is done in every sampling period employing the on-line version of the back-propagation method [62], [84]. This enables the controller to track the plant variations as they occur to yield the optimum performance. The on-line training process comprises the following steps:

1) At time step k, $y(k)$ is sampled.

2) Using $y(k)$ and $\hat{y}(k)$, the weights of the ANI are updated, minimizing $J_i(k)$.

3) The output of the controller, $u(k)$, is computed.

4) Using $u(k)$, the predicted plant output, $\hat{y}(k+1)$, is computed by the ANI.

5) Based on $\hat{y}(k+1)$, the weights of the ANC are updated, minimizing $J_c(k)$.

In step 2 above, the training is straightforward since the error at the output of the ANI is known. However, in step 5 the training is not as easy, since the error at the output of the ANC is not known. In this case, first the weights of the ANI are frozen and the error between the desired and predicted plant output is back-propagated through the ANI. Then, the back-propagated signal at the input of the ANI is further back-propagated through the ANC, making the necessary changes to the controller weights. In other words, for adapting the weights of the controller, the identifier acts as a channel to convey the error from the output of the identifier to the output of the controller. This justifies the need to have the identifier.

The error used to train the ANI and the ANC are both scalar and the learning is done only once in each sampling period for each of the two subnetworks. This simplifies the training algorithm in terms of computation time, which is of special importance in real-time implementation.

3.5 Summary

In this chapter various plant models and identification models are discussed. Based on the discussion, a plant model and identification model are chosen to be used for the implementation of neural adaptive power system stabilizer presented in the next part. An indirect adaptive controller based on on-line trained neural networks is also introduced in this Chapter. The proposed controller consists of two subnetworks; an adaptive neuro-identifier and an adaptive neuro-controller. These two subnetworks are trained in on-line mode using the back-propagation method. Details of the training procedure are also explained. The on-line training process enables the controller to track the variations in control environment and act accordingly. It also considers the non-linear nature of the plant. Using the scalar error vector is another advantage of this algorithm. This reduces the computation burden of the adaptive on-line algorithm.

Part II

Simulation Studies

Chapter 4

On-Line Identification of Synchronous Generator Using Neural Networks

4.1 Introduction

In the previous chapter, different plant and identification models were studied. In order to verify the suitability of the plant and identifier models chosen for the identification of synchronous generator, a simulation study should be conducted. In this chapter, plant Model IV and the series-parallel identifier model are combined in order to build an on-line trained neuro-identifier. The proposed identifier has a simple architecture and is trained using the back-propagation method in on-line mode. It is verified in a variety of operating conditions and disturbances. Simulation results confirm the suitability of the models used and demonstrates the effectiveness of the identifier in tracking the synchronous generator.

4.2 Identifier Structure

The identifier structure is studied in this section. The popular MLP network with back-propagation learning has been used to develop the identifier. A variety of structures were tested for the identifier. Different number of inputs (i.e. from four to twelve), hidden layers (i.e. one and two) and hidden neurons (i.e. from four to sixteen) were tested. The network with 4 inputs, which uses two signals and their delays, did not generate good results, regardless of the number of hidden neurons. The 6x8x1 network generated good results for different tests. The networks larger than that did not improve the result. Therefore, the 6x8x1 structure, as shown in Fig. 4.1, was chosen for the identifier. Sigmoid non-linearity was used for the hidden neurons. The output neuron was chosen to have linear characteristics. The input vector to

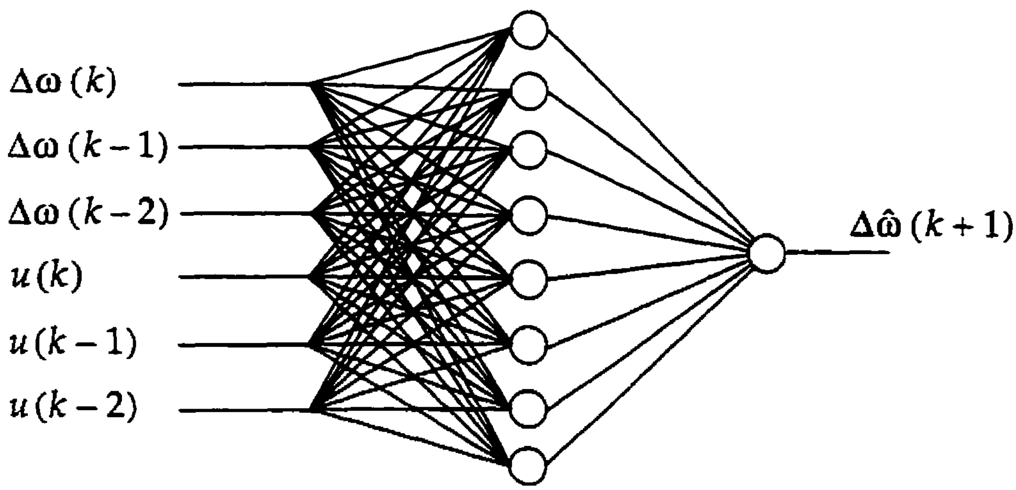


Figure 4.1: The neuro-identifier.

the neuro-identifier is

$$[\Delta\omega(k), \Delta\omega(k-1), \Delta\omega(k-2), u(k), u(k-1), u(k-2)] \tag{4.1}$$

where $\Delta\omega(k)$ is the generator speed deviation and $u(k)$ is the power system stabilizer (PSS) control signal (generator input), both at the time step k. The output of the identifier is the predicted speed deviation at the time step $k+1$, $\Delta\hat{\omega}(k+1)$.

The inputs to the neuro-identifier are scaled before being applied to the network to take a value in the range of $[-1, +1]$. This is because of the fact that the sigmoid non-linearity used changes between these two values. This also makes the weights of the first layer not to take very large values. The cost function defined for the identifier is:

$$J_i(k) = \frac{1}{2}[\Delta\omega(k) - \Delta\hat{\omega}(k)]^2 \tag{4.2}$$

The identifier goes through two stages of training, namely off-line and on-line training. In off-line training, first the identifier is trained using the input-output data for a variety of operating conditions and disturbances. The operating condition changes in the range of 0.1 *pu* to 1.0 *pu* power output and 0.7 pf lead to 0.1 pf lag. The disturbances used were the voltage reference and input torque reference disturbances and three phase to ground fault. The training was iteratively done until a pre-specified tolerance is met.

After the off-line training, the network is further trained on-line. The cost function 4.2 is minimized using back-propagation method in the on-line mode. At each sampling instant, the input and the output of the generator are sampled and the input vector to the identifier is formed as in (4.1). Then the error between the output of the plant, i.e., desired output, and the identifier, which is a scalar value, is back-propagated through the identifier to make necessary updates to the weights of the network. This process is repeated every sampling period making the training on-line, which in turn results in an adaptive approach to identify a plant.

4.3 Simulation Results

The performance of the proposed identifier is investigated on a synchronous generator connected to a constant voltage bus through two parallel transmission lines as shown in Fig. 4.2. A non-linear seventh-order model is used to simulate the dynamic behavior of the single-machine infinite-bus power system. The differential equations used to simulate the synchronous generator, the transfer function of the governor, AVR and CPSS along with the system parameters are given in Appendix A. A sampling rate of 25 Hz is chosen for the digital system. The response of the identifier after training is compared to that of the plant for various disturbances under different operating conditions as explained in the following subsections.

4.3.1 Loaded Generator

With the generator operating at P_e=0.7 pu, pf=0.85 lag a 0.05 pu step increase in input torque reference is applied at 1 s. The generator speed deviation and its predicted value are shown in Fig. 4.3. The figure clearly shows the effectiveness of the identifier in tracking the generator.

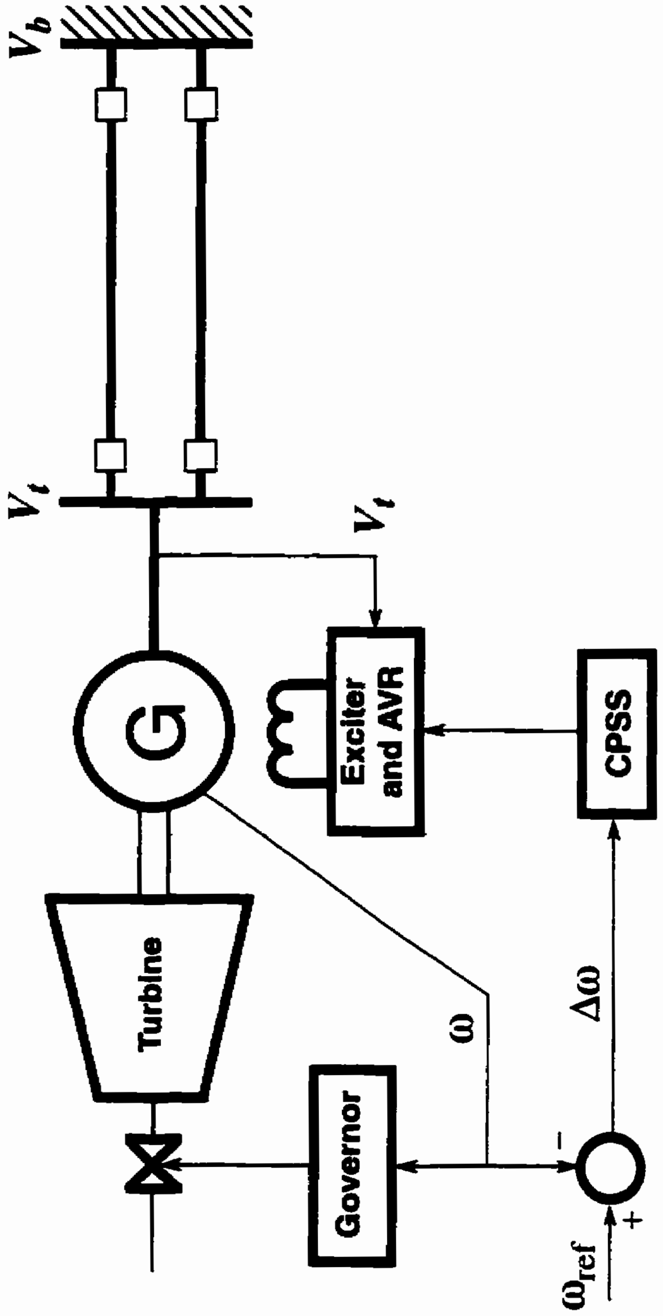


Figure 4.2: Power system model configuration.

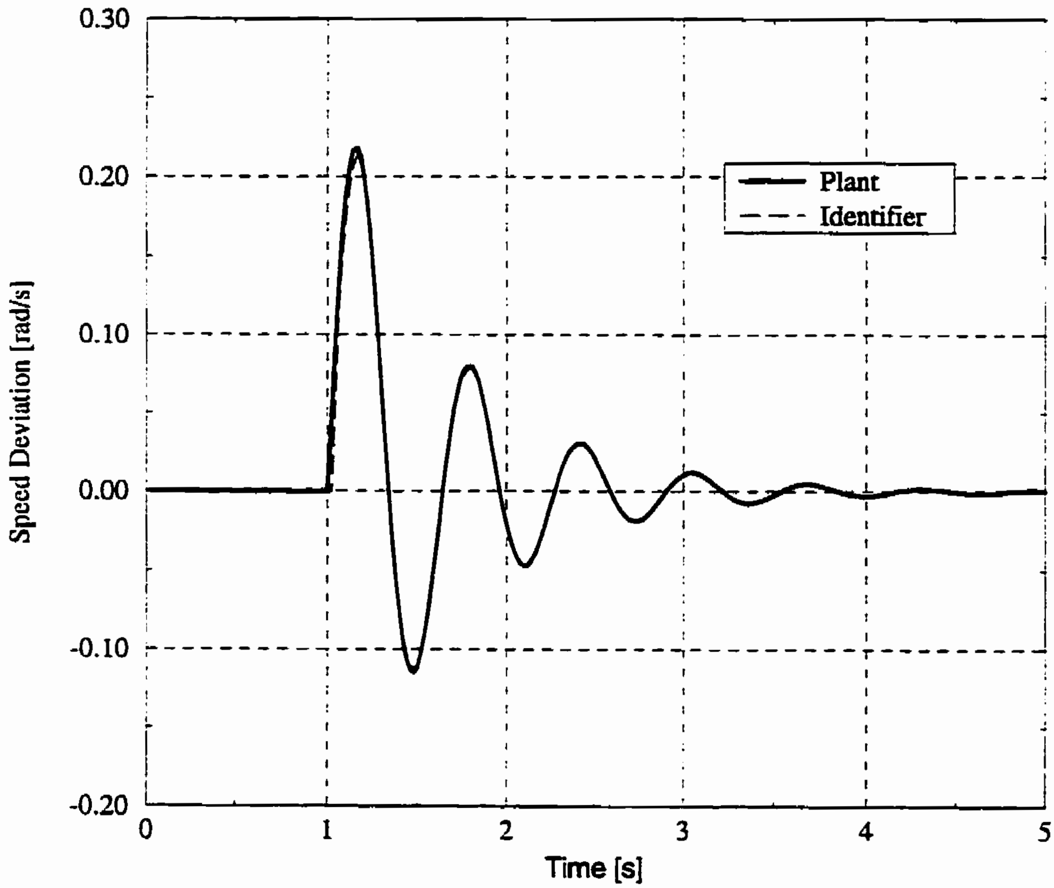


Figure 4.3: Plant and identifier responses to a 0.05 *pu* step increase in torque.

4.3.2 Light Load

In light load test, the generator is operating at P_e=0.2 *pu*, pf=0.85 lag when a 0.15 *pu* step increase in input torque reference is applied at 1 *s*. Fig. 4.4 shows the response of the system under the new operating condition and disturbance. Again, very good tracking is achieved using the neuro-identifier.

4.3.3 Leading Power Factor

In this test, the generator is operating under a leading power factor condition. A 0.10 *pu* step increase in input torque reference is applied at 1 *s* while generator is operating at P_e=0.3 *pu*, pf=0.9 lead. Fig. 4.5 depicts the plant and identifier responses to such a disturbance.

4.3.4 Voltage Reference Change

In this test, the ability of the identifier in tracking the generator when a voltage disturbance occurs is verified. A 0.05 *pu* step increase in exciter reference voltage is applied at 1 *s* with the generator operating at P_e=0.2 *pu*, pf=0.85 lag. Result given in Fig. 4.6, shows that the proposed identifier can track the plant satisfactorily.

4.4 Summary

An on-line trained identifier to track synchronous generator is introduced in this chapter. The proposed identifier uses the feed-forward multi-layer neural networks. Its structure is very simple and there is no need for a

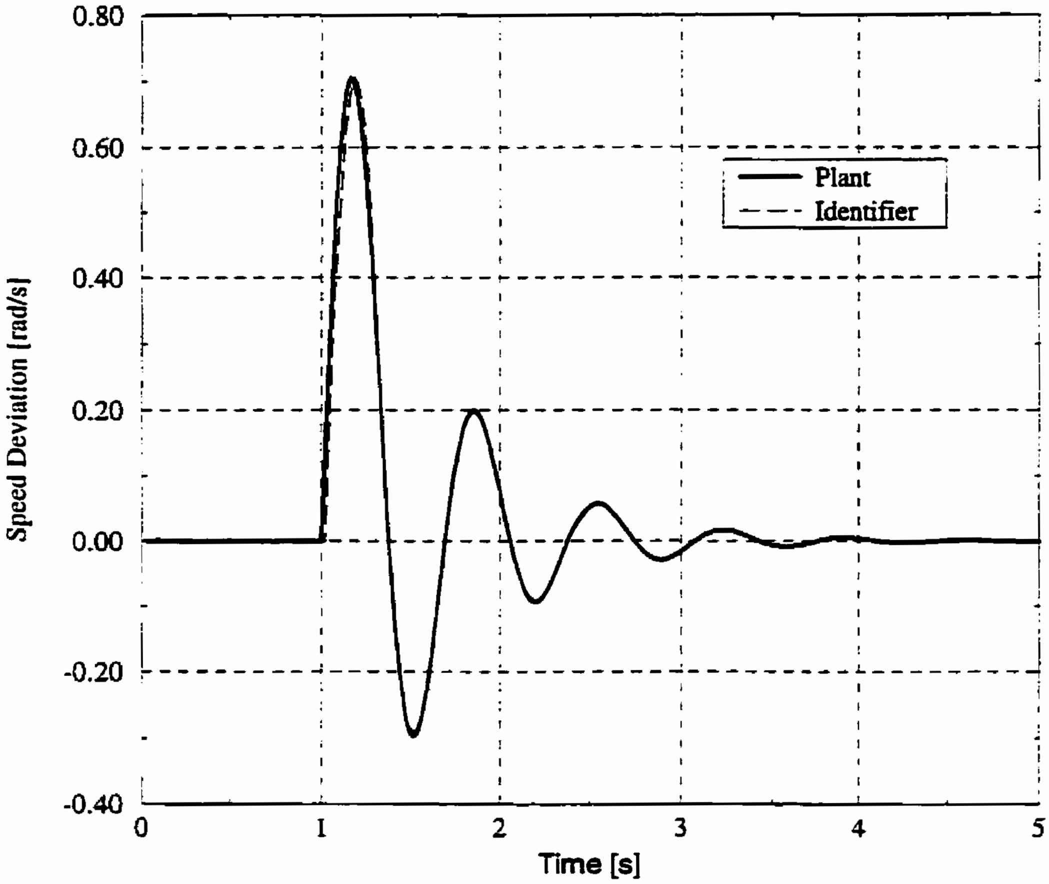


Figure 4.4: Plant and identifier responses to a 0.15 *pu* step increase in torque.

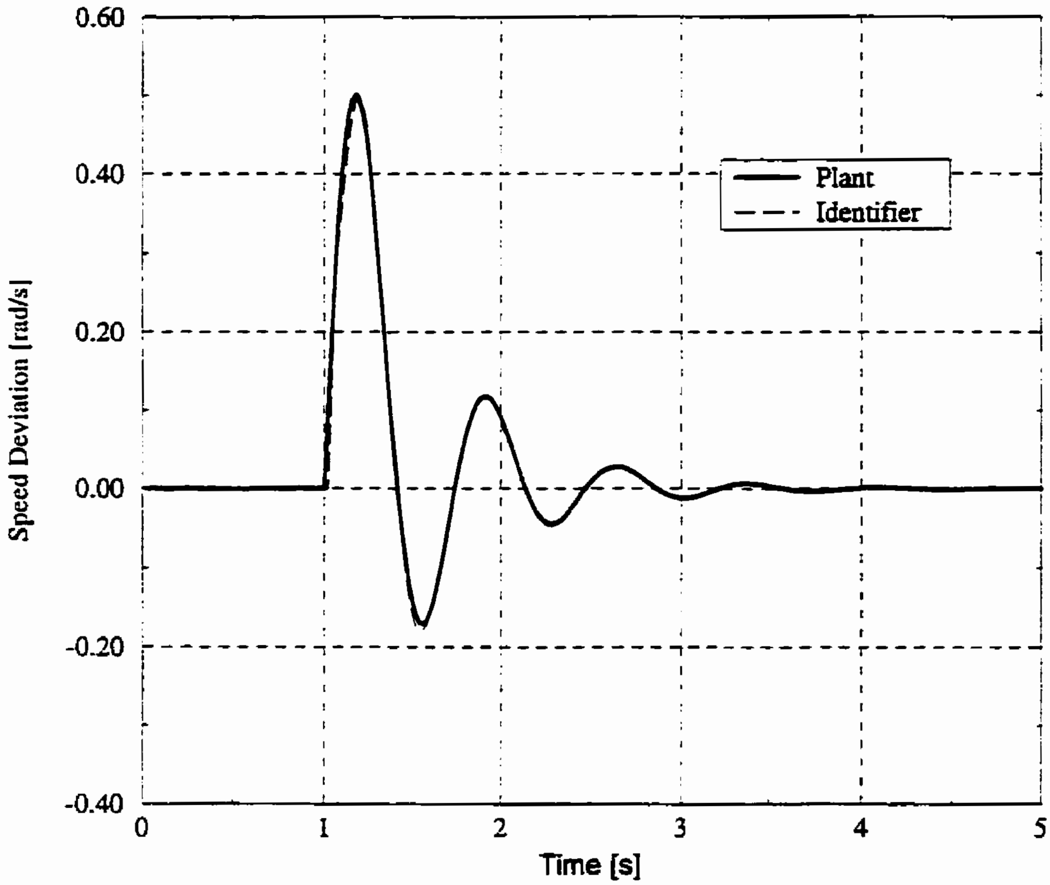


Figure 4.5: Plant and identifier responses to a 0.10 *pu* step increase in torque.

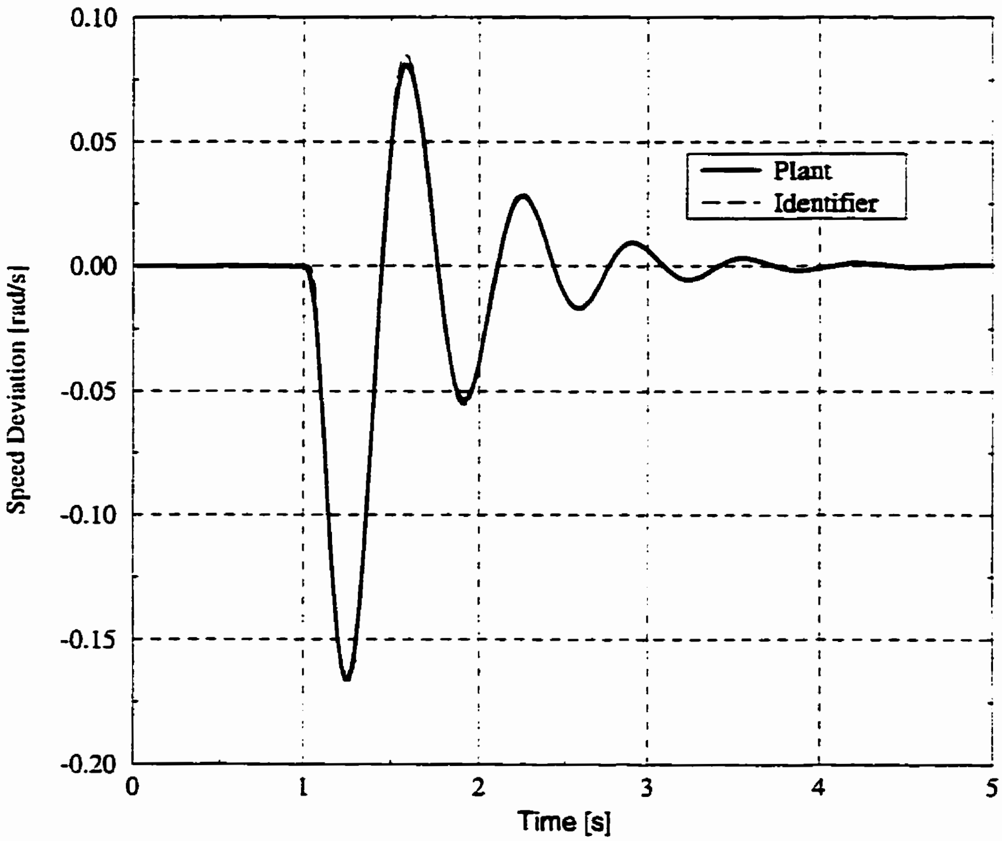


Figure 4.6: Plant and identifier responses to a 0.05 *pu* step increase in exciter reference voltage.

large number of neurons in its implementation. The training algorithm is simplified by making use of a single element error vector. The simulation results show the effectiveness of the tracking ability of the proposed identifier under various operating conditions and disturbances.

Chapter 5

NAPSS Application in Single-Machine Power System

5.1 Introduction

Low frequency oscillations are a common problem in large interconnected power systems [90]. Power system stabilizer (PSS) can provide supplementary control signal to the excitation system and/or governor system of the electric generating unit to damp out these oscillations and to improve generator's dynamic performance [9], [91]. Conventional power system stabilizer (CPSS) is a lead-lag compensation-type device, based on linear control theory [8]. It has been adopted by most utility companies because of its simple structure, flexibility and ease of implementation, and it has made a great contribution in enhancing power system damping and dynamic stability [37].

The CPSS parameters are tuned based on the linear model of the power system. After off-line tuning of the parameters, extensive field testing is done at the time of commissioning. However, power systems are highly non-linear systems. Moreover, their configuration and parameters change with time. In fact, it has been found that the dynamic properties of the power system are quite different for different operating conditions [92]. This brings discrepancies between the mathematical linear model of the power system and the physical non-linear plant. Therefore, the parameters of the CPSS must be retuned so that it can continue to provide the desired performance. Even under nominal operating conditions, there is still some uncertainty due to the approximate knowledge of the power system parameters. Thus, to yield satisfactory control performance, it is desirable to develop a stabilizer which considers the non-linear nature of the plant and has the ability to adjust its own parameters on-line according to the environment in which it is working.

With the development of power systems and the increasing demand for quality electricity supply, modern control techniques are being investigated. In recent years, there have been new approaches to PSS design using modern control techniques [46], [50], [93]. Having a variety of advantages, neural networks have also been applied to power system control problems. In [94], a neural network regulator for a turbogenerator was proposed to control the voltage and speed of the generator. Design of a PSS based on neural networks was also suggested in [95]. In that paper, the authors employed an external teacher (a non-linear controller based on variable structure control theory) to train the neuro-PSS. This way, the neural network is used to realize a complicated control algorithm in a comparatively easy way. In [96], a sophisticated training algorithm was proposed for the neuro-PSS. In [97], Zhang proposed a few off-line methods to design a neuro-PSS. First, he designed the neuro-PSS by employing the pole-shifting adaptive PSS as a teacher. Then, he proposed an inverse I/O mapped Neuro-PSS to be trained directly from plant I/O data. In the final design, he developed a multi-input Neuro-PSS in which he used both speed deviation and electrical power deviation. But, as indicated earlier, all of his designs were off-line and there was no on-line adaptation involved. An on-line trained neuro-control system for power system stabilization was also proposed in [98]. There, the authors proposed a neural-network based PSS using two feed-forward networks which needs the measurement of all the generator states. The proposed PSS does not use tapped delay elements to consider dynamic characteristics of the generator. The authors then verified their PSS on a single-machine infinite-bus system employing a third-order model for the generator.

In this Chapter, the neural adaptive power system stabilizer (NAPSS) is presented. Then, it is applied to the single-machine infinite-bus power system. The control architecture consists of two neural networks; the adaptive neuro-identifier (ANI) to track the plant, and the adaptive neuro-controller (ANC) to damp out the output oscillations. Using the on-line training method, the NAPSS is able to track the plant variations as they occur and to provide the control signal accordingly. It has a simple structure and does not require the internal states of the plant. It is trained directly from the output performance and does not need any reference model or teacher signal. The performance of the proposed NAPSS under different load conditions and disturbances is investigated for the single-machine infinite-bus system.

5.2 Single-Machine Power System Model

A nonlinear seventh-order model is used to simulate the dynamic behavior of the generating unit connected to a constant voltage bus through two parallel transmission lines. A schematic diagram of the system is shown in Fig. 5.1. For comparison purposes, the CPSS is also included in the studies. The differential equations used to simulate the synchronous generator, the transfer function of the governor, AVR and CPSS along with the system parameters are given in Appendix A. Studies performed with various sampling rates show that the performance is practically the same for a sampling rate in the range of 20-100 *Hz*. Sampling frequencies above 100 *Hz* are of no practical benefit and the performance deteriorates for sampling rates under 20 *Hz*. A sampling rate of 25 *Hz* has been chosen to make sure that there is enough time available for weight update calculations.

5.3 Controller Structure

The structure of the controller for single-machine study is shown in Fig. 5.2. It consists of two subnetworks. The first subnetwork is an adaptive neuro-identifier (ANI) which tracks the dynamic behavior of the plant and identifies the plant in terms of its internal weights, and the second one is an adaptive neuro-controller (ANC) to provide the necessary control action so as to damp out the oscillations of the plant output.

The input vector to the ANI is

$$[\Delta\omega(k), \Delta\omega(k-1), \Delta\omega(k-2), u(k), u(k-1), u(k-2)] \tag{5.1}$$

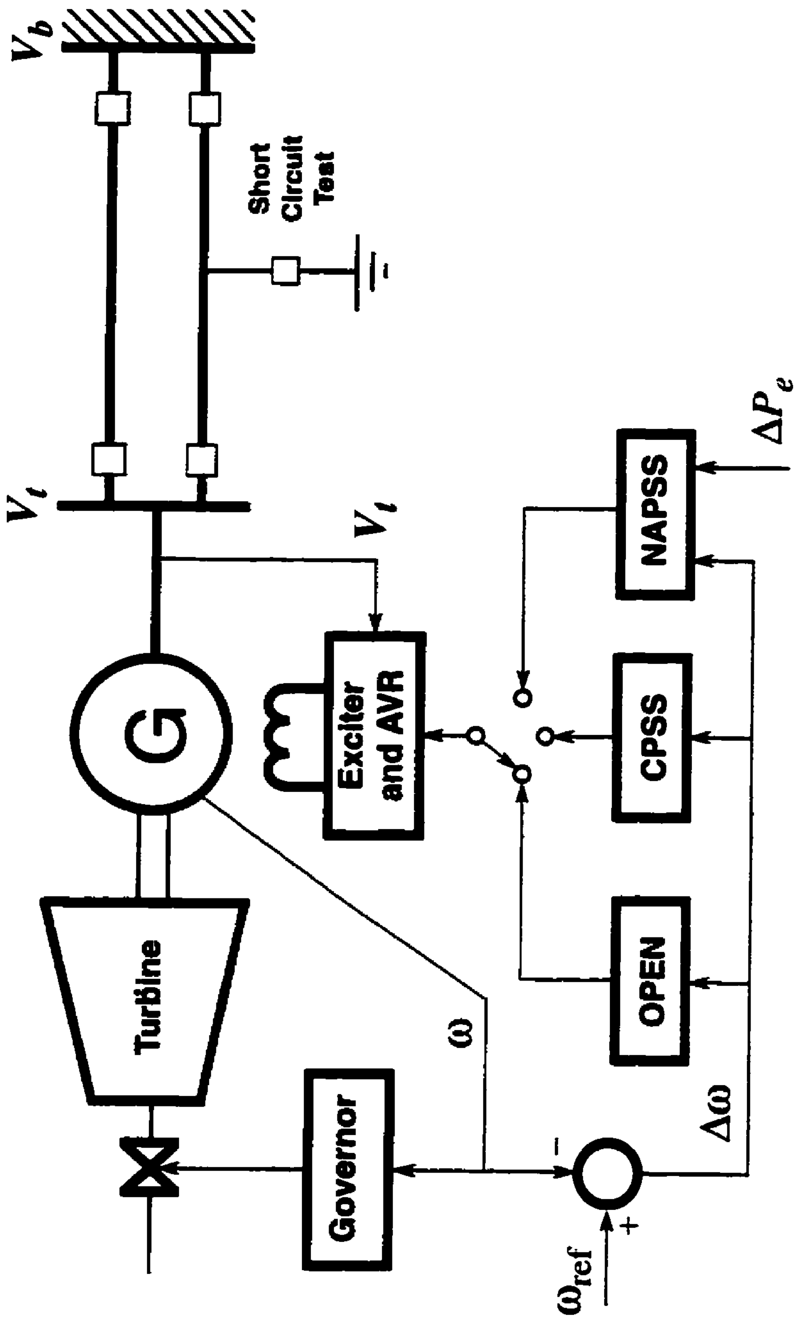


Figure 5.1: Power system model configuration.

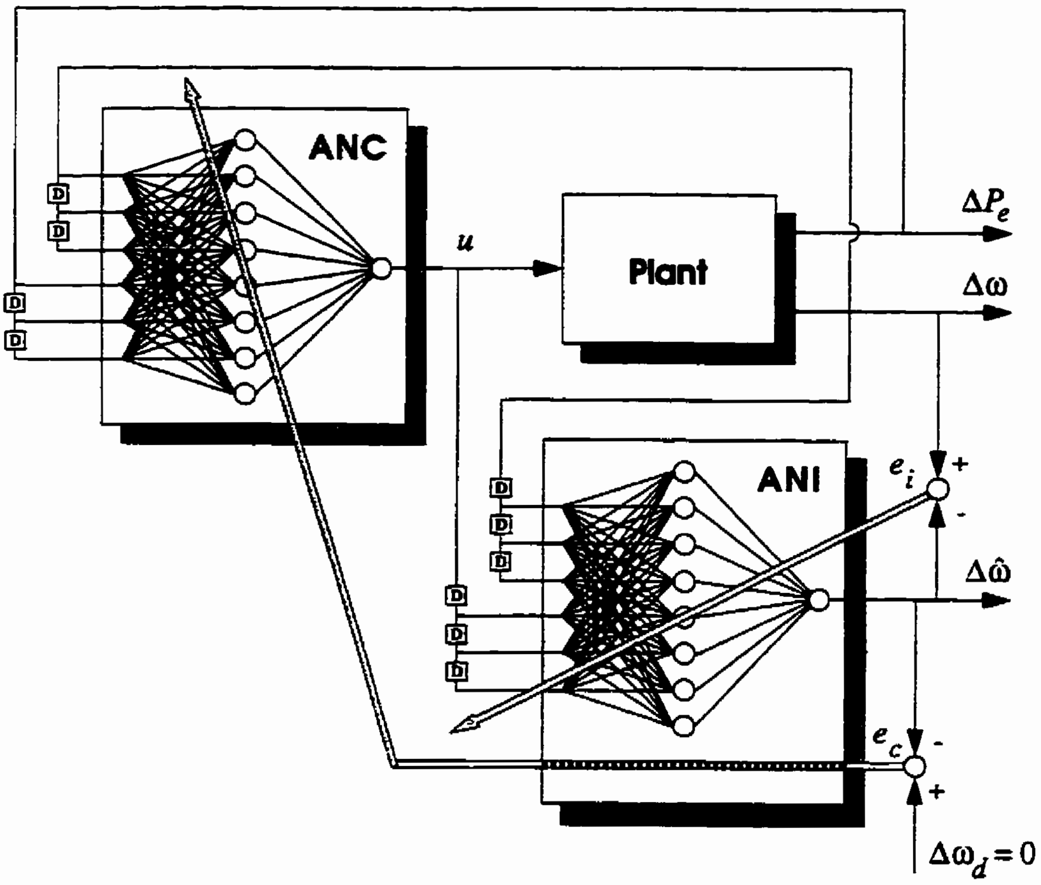


Figure 5.2: Controller structure for single-machine study.

where $\Delta\omega(k)$ is the generator speed deviation and $u(k)$ is the PSS output (generator input), both at time step k. The output of the identifier is the predicted speed deviation, $\Delta\hat{\omega}(k+1)$, at time step $(k+1)$. The input vector to the ANI is scaled before being applied to the network to take a value in the range of $[-1, +1]$. The cost function used for the ANI is

$$J_i(k) = \frac{1}{2}e_i(k)^2 = \frac{1}{2}[\Delta\omega(k) - \Delta\hat{\omega}(k)]^2 \tag{5.2}$$

The weights are updated as

$$W_i(k) = W_i(k-1) - \eta_i \nabla_{W_i} J_i(k) \tag{5.3}$$

in which $W_i(k)$ is the matrix of identifier weights at time step k and η_i is the learning rate for the ANI. The gradient $\nabla_{W_i} J_i(k)$ is computed by

$$\nabla_{W_i} J_i(k) = -[\Delta\omega(k) - \Delta\hat{\omega}(k)]\frac{\partial\Delta\hat{\omega}(k)}{\partial W_i(k)} \tag{5.4}$$

in which $\frac{\partial\Delta\hat{\omega}(k)}{\partial W_i(k)}$ is a vector of partial derivatives of $\Delta\hat{\omega}(k)$ with respect to each element of $W_i(k)$. Using (5.3) and (5.4), the cost function $J_i(k)$ is minimized in each sampling period by back-propagating the scalar error $[\Delta\omega(k) - \Delta\hat{\omega}(k)]$.

The input vector to the ANC is:

$$[\Delta\omega(k), \Delta\omega(k-1), \Delta\omega(k-2), \Delta P_e(k), \Delta P_e(k-1), \Delta P_e(k-2)] \tag{5.5}$$

where $\Delta P_e(k)$ is the accelerating power at time step k. The output of the ANC is the PSS control action, $u(k)$, at time step k. The inputs to the ANC are also scaled in the range of $[-1, +1]$. The cost function for the ANC is considered as:

$$\begin{aligned} J_c(k) &= \frac{1}{2}[e_c(k)^2 + hu(k)^2] \\ &= \frac{1}{2}[\Delta\omega_d(k) - \Delta\hat{\omega}(k)]^2 + \frac{h}{2}u(k)^2 \end{aligned} \tag{5.6}$$

where $\Delta\omega_d(k)$ is the desired speed deviation at time step k, which is equal to zero in a regulatory setup, and h is a tuning parameter which is used to improve the plant output dynamic characteristics. By taking h greater than zero, a penalty factor is applied to the control action generated by the NAPSS which helps the tuning of the dynamic trajectory and optimizing the overshoot and the settling time of the response curve. The controller weights, $W_c(k)$, are updated as

$$W_c(k) = W_c(k-1) - \eta_c \nabla_{W_c} J_c(k) \tag{5.7}$$

where η_c is the controller learning rate and the gradient $\nabla_{W_c} J_c(k)$ is defined as

$$\nabla_{W_c} J_c(k) = [\Delta\hat{\omega}(k)\frac{\partial\Delta\hat{\omega}(k)}{\partial u(k)} + hu(k)]\frac{\partial u(k)}{\partial W_c(k)} \tag{5.8}$$

Using (5.7) and (5.8), $J_c(k)$ is minimized each sampling period. As it is seen in (5.1) and (5.5), the generator states are not required for the implementation of the ANI and ANC and only input-output data are used. This greatly simplifies the implementation of the NAPSS.

5.4 Training Process

The success of the control algorithm presented in section 5.3 depends highly on the accuracy of the identifier in tracking the dynamic plant. If the identifier is not sufficiently trained, the control signal which is computed based on the identifier parameters can not be trusted and may result in unsatisfactory response. For this reason, the ANI is initially trained off-line before being hooked up in the final configuration. The training is performed over a wide range of operating conditions and a wide spectrum of possible disturbances for the generating unit. It is further discussed in the next section. After the off-line training stage, the ANI is hooked up in the system. Further training of the ANI and ANC is done in every sampling period employing the on-line version of the back-propagation method [62]. This enables the NAPSS to track the plant variations as they occur to yield the optimum performance. The training of the NAPSS comprises the following steps:

1) At time step k, $\Delta\omega(k)$ and $\Delta P_e(k)$ are sampled.

2) Using $\Delta\omega(k)$ and $\Delta\hat{\omega}(k)$, the weights of the ANI are updated, minimizing $J_i(k)$.

3) The output of the controller, $u(k)$, is computed.

4) Using $u(k)$, the predicted speed deviation, $\Delta\hat{\omega}(k+1)$, is computed by the ANI.

5) Based on $\Delta\hat{\omega}(k+1)$, the weights of the ANC are updated, minimizing $J_c(k)$.

In step 2 above, the training is straightforward since the error at the output of the ANI is known. However, in step 5 the training is not as easy, since the error at the output of the ANC is not known. In this case, first the weights of the ANI are frozen and the error between the desired and the predicted plant output is back-propagated through the ANI. Then, the back-propagated signal at the input of the ANI is further back-propagated through the ANC, making the necessary changes to the controller weights. The MATLAB function to perform the controller training is given in Appendix B. The error used to train the ANI and the ANC are both scalar and the learning is done only once in each sampling period for each of the two subnetworks. This simplifies the training algorithm in terms of computation time, which is of special importance in real-time implementation.

Moreover, in order to further simplify the training algorithm, the use of dynamic back-propagation method [99], [100], [101], which considers the tapped-delay elements present in the input of the ANI, is avoided here due to the extra computation burden involved. Static back-propagation [78] has been used instead, since it is accurate enough for the purpose of this application. The parameters of the identifier and controller along with those of the learning algorithm are discussed in next section.

5.5 Simulation Studies

5.5.1 Parameter Setting

A variety of structures were tested for the ANC. Different number of inputs (i.e. from four to twelve), hidden layers (i.e. one and two) and hidden neurons (i.e. from four to sixteen) were tested. The network with 4 inputs, which uses two signals and their delays, did not generate good results, regardless of the number of hidden neurons. The 6x8x1 network generated good results for different tests. The networks larger than that did not improve the result. Therefore, the 6x8x1 structure was chosen for the ANC. The ANI structure is also 6x8x1 as explained in the previous Chapter. For both networks, the hidden neurons have sigmoid nonlinearity and the output neuron is linear.

Initial weights of the ANC lie between $[-0.1, +0.1]$, chosen randomly at the beginning of the process. The initial weights of the ANI are set to those obtained from off-line training stage of the ANI as discussed before. The learning rate for the ANI and the ANC is 0.01 and 0.03, respectively. The value of h, the tuning parameter, is set to 4.5. It is worth mentioning that all of the above mentioned parameters as well as the network structure were found through simulation and trial and error. For off-line training, data were collected for operating conditions in the range of 0.1 *pu* to 1.0 *pu* power output and 0.7 pf lead to 0.1 pf lag. The training was iteratively done until a pre-specified tolerance is met. The disturbances used were the voltage reference and input torque reference disturbances and three-phase to ground fault.

5.5.2 CPSS Parameter Tuning

With the generator operating at a power output of 0.7 *pu*, 0.85 pf lag, a 0.05 *pu* step increase in input torque reference is applied at time 1 s. At time 5 s, the change in torque reference is removed and the system returns to its original operating condition.

Under the above conditions, the CPSS was carefully tuned for the best possible performance, i.e. the overshoot and the settling time were minimized. The parameters of the CPSS were then kept unchanged for all of the tests performed. Results of the study with the NAPSS, the CPSS and without a stabilizer are shown in Fig. 5.3. It is seen from the figure that the NAPSS damps out the low frequency oscillations very quickly.

5.5.3 Loaded Generator Test

In this test, a 0.10 *pu* step increase in input torque reference is applied at 1 *s* and removed at 5 *s*. The generator is operating at 0.7 *pu* power, 0.85 *pf* lag. The system response given in Fig. 5.4 shows that the NAPSS can handle the disturbance better than the CPSS and the oscillations settle down more quickly. This test demonstrates the effectiveness of the NAPSS in damping the low frequency oscillations.

5.5.4 Light Load Test

The system condition is the same as in the previous case except that the generator is now operating under a light load condition, i.e. 0.2 *pu* power, 0.85 *pf* lag. The disturbance is a 0.05 *pu* step decrease in the voltage reference. Fig. 5.5 shows the result of the system with the CPSS and the NAPSS. It is evident that, despite a large change in the operating conditions, the NAPSS still provides good result because of the adaptation process.

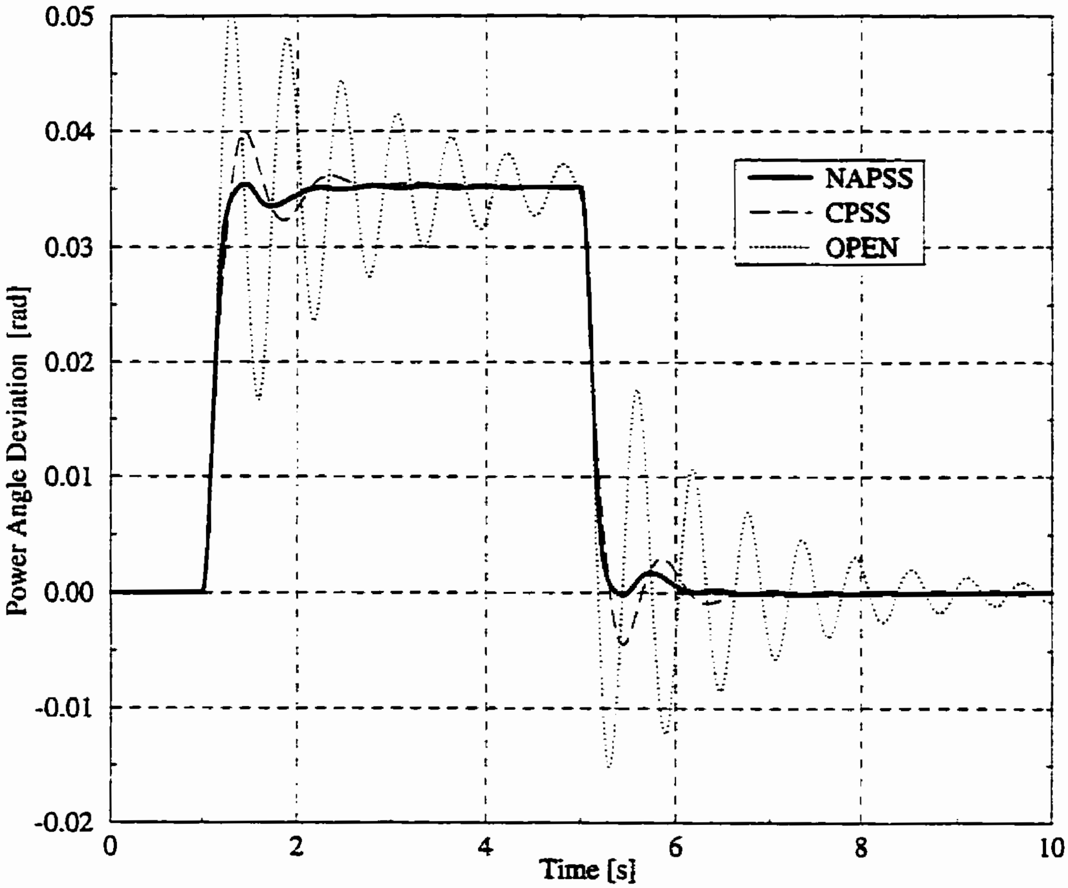


Figure 5.3: System response to a 0.05 *pu* step increase in torque and return to original condition.

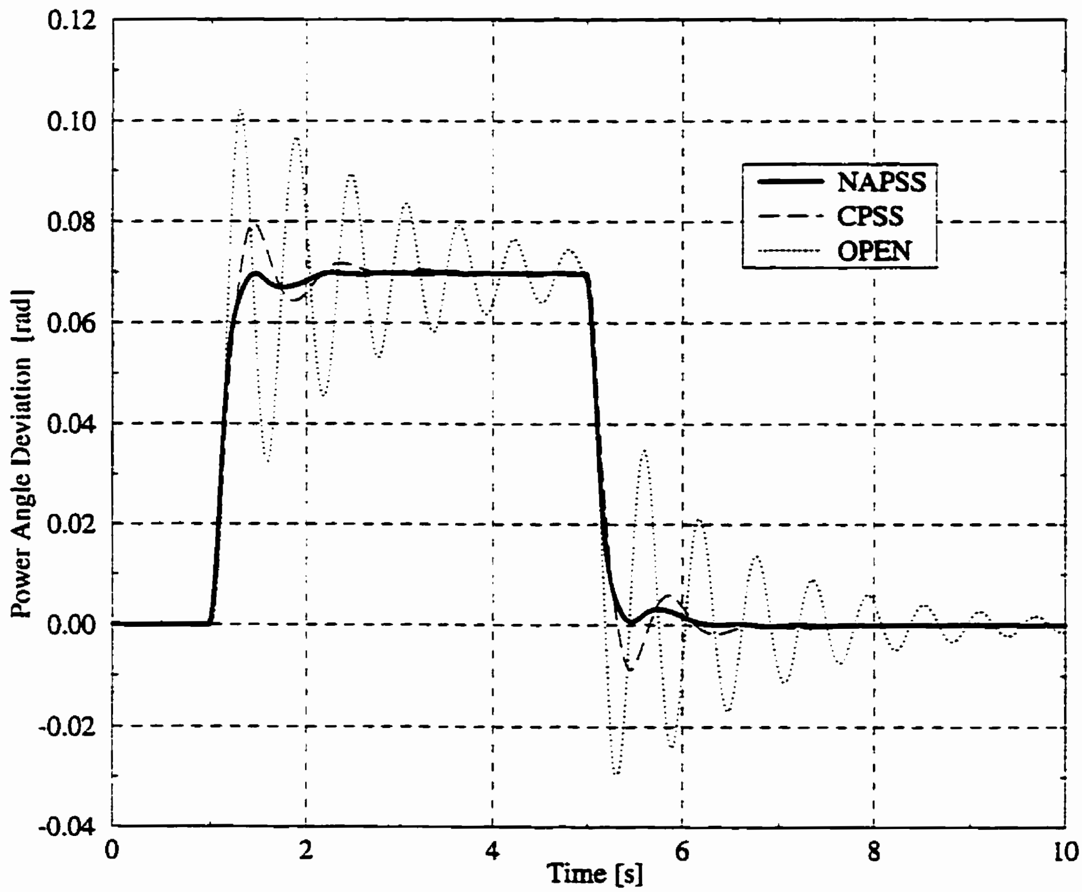


Figure 5.4: System response to a 0.10 *pu* step increase in torque and return to original condition.

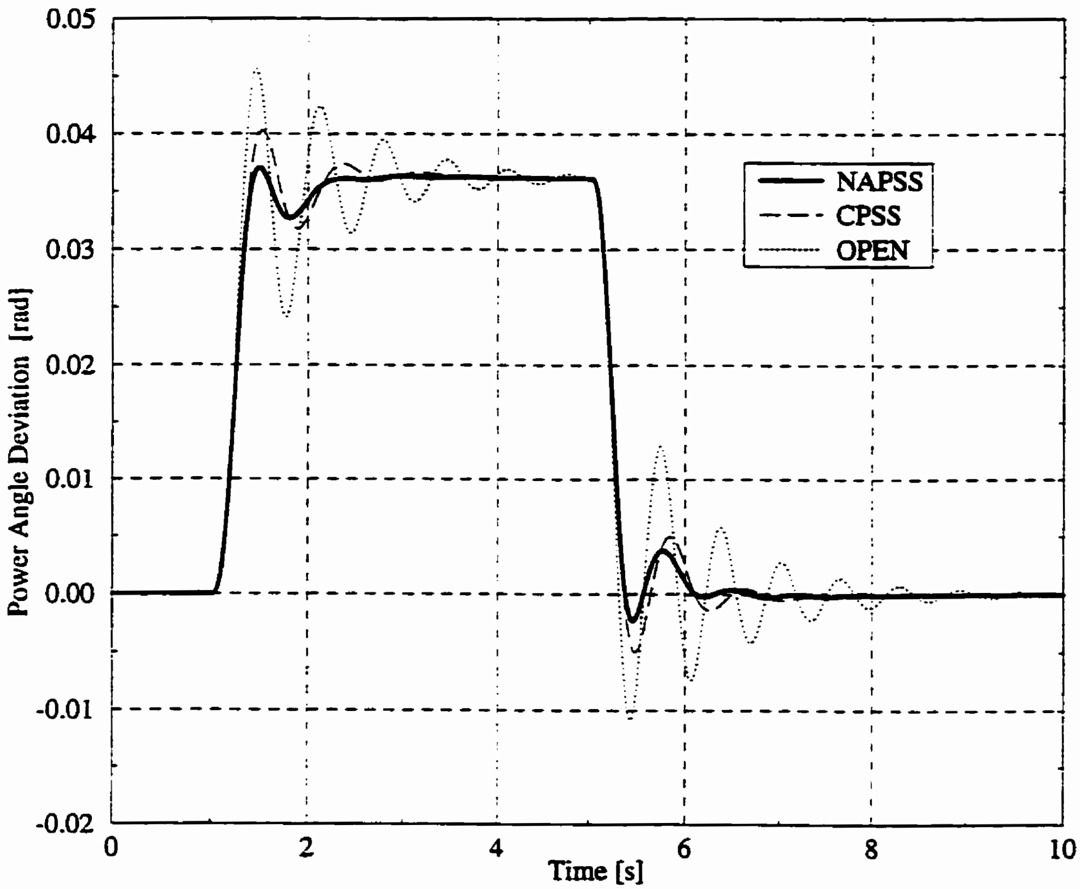


Figure 5.5: System response to a 0.05 *pu* step decrease in voltage reference and return to original condition in light load test.

5.5.5 Leading Power Factor Operation Test

When the generator is operating at a leading power factor, it makes the situation difficult for the controller because the stability margin is reduced. However, in order to absorb the capacitive charging current in a high voltage power system in light load condition, it may sometimes become necessary to operate the generator at a leading power factor. Thus, it is desirable that the controller be able to guarantee stable operation of the generator under a leading power factor condition.

A test is conducted with the generator at 0.3 *pu* power, 0.9 *pf* lead. A disturbance of 0.20 *pu* step increase in the input torque reference was applied. This disturbance is high enough to cause the system to operate in the nonlinear region. The results given in Fig. 5.6 show that the oscillation of the system is damped out quickly and demonstrates the effectiveness of the NAPSS to control the generator under leading power factor operating conditions. The control signals of both NAPSS and CPSS are given in Fig. 5.7.

5.5.6 Fault Test

To verify the behavior of the proposed neural adaptive stabilizer under transient conditions, a fault is applied to the system. For this study, the equivalent reactance of the double circuit transmission line was set at 0.4 *pu* instead of 0.6 *pu*. The response of the power system to a three-phase to ground short circuit at the middle of one transmission line, cleared 200 *ms* later by the disconnection of the faulted line and successful reclosure after 4 *s* is shown in Fig. 5.8. It can be seen that the NAPSS minimizes the deviation of the

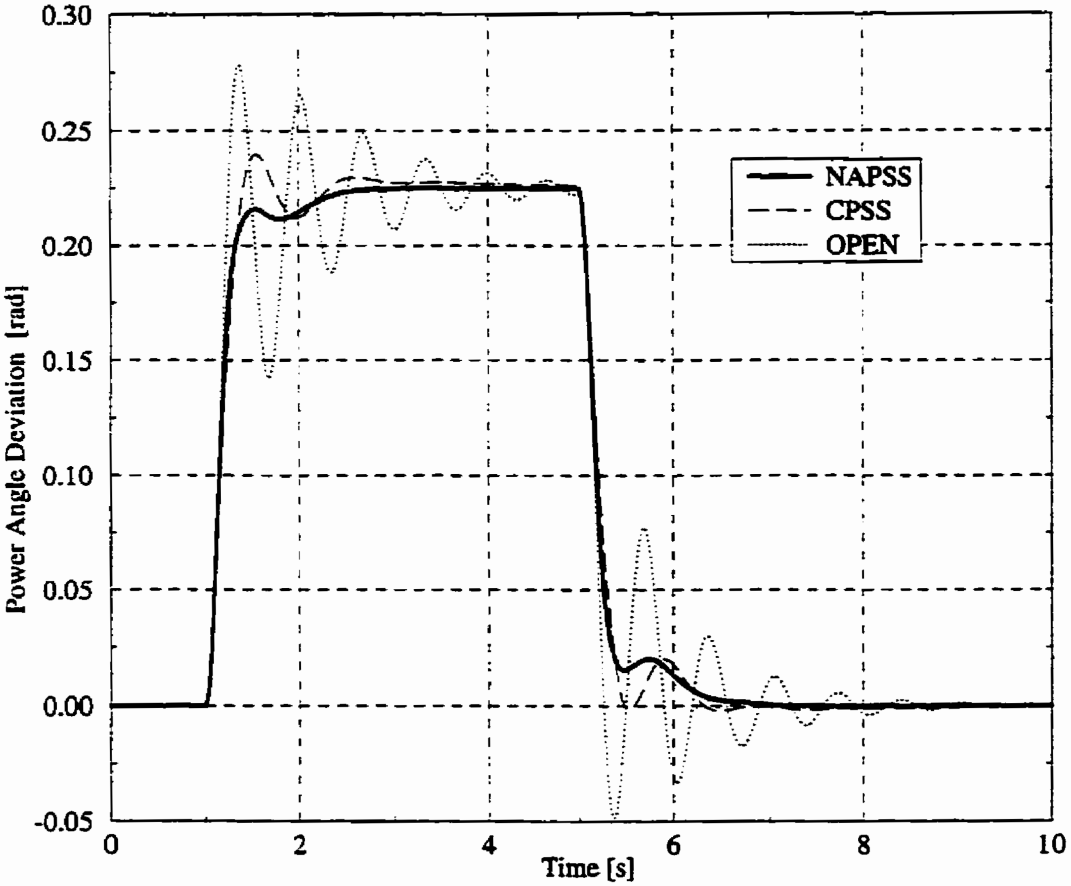


Figure 5.6: System response to a 0.20 *pu* step increase in torque and return to initial condition in leading power factor test.

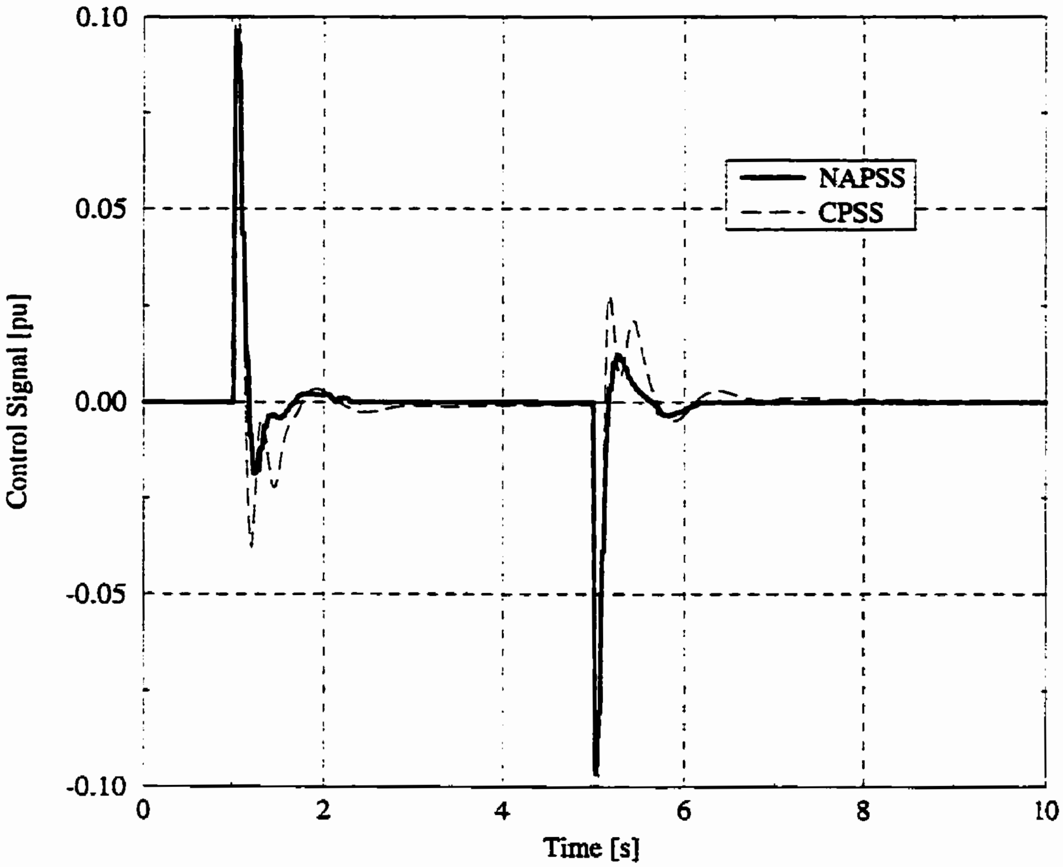


Figure 5.7: Control signals of NAPSS and CPSS for a 0.20 *pu* step increase in torque and return to initial condition in leading power factor test.

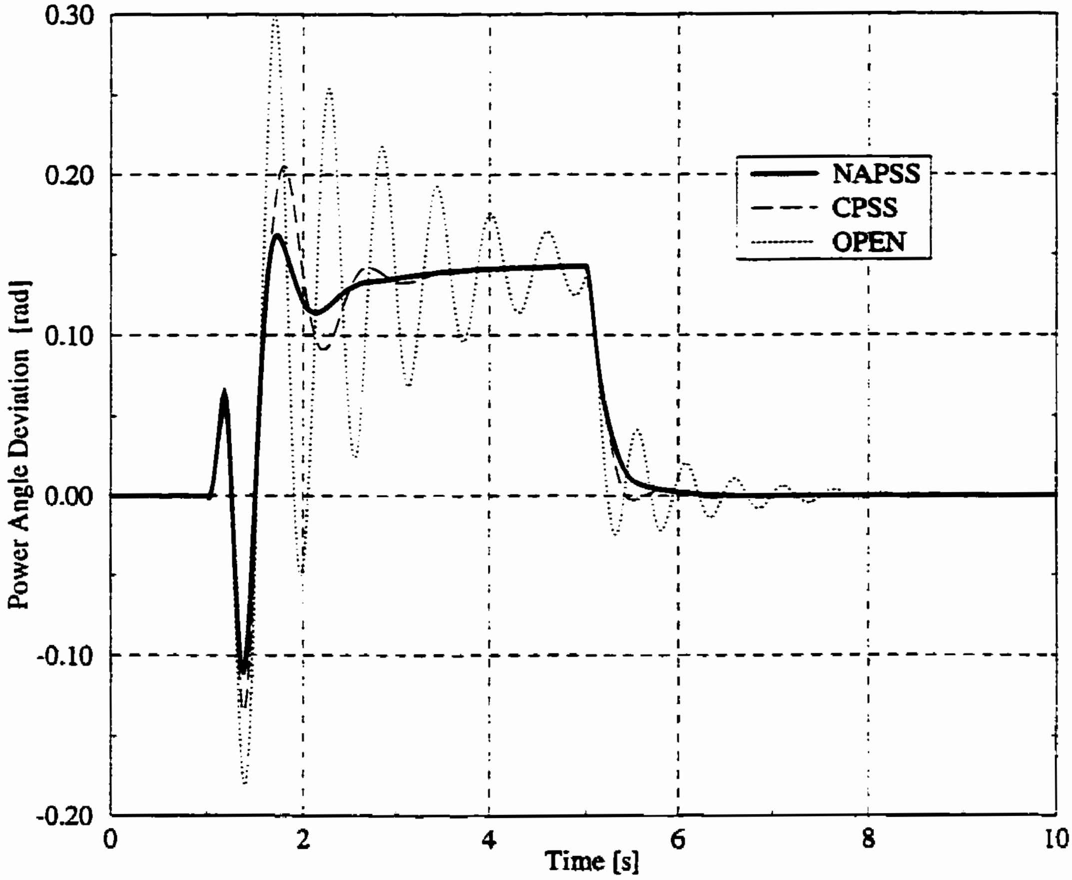


Figure 5.8: System response to a three-phase to ground fault at the middle of one transmission line.

power angle of the generator after the fault and helps the system to reach the new operating point very quickly. The test shows that the NAPSS not only improves the dynamic performance but also enhances the transient performance of the system. An important fact worth mentioning here is that the parameters of the NAPSS were not required to be tuned for different test conditions. This indicates that the proposed stabilizer enjoys the high adaptability to the operating conditions.

5.5.7 Different Line Impedances Test

The parameters of the CPSS have to be re-tuned if the configuration and/or parameters of the power system change. Otherwise, its performance cannot be guaranteed. However, with the Neural Adaptive PSS, since the controller is adapted on-line based on the output performance, the control algorithm can automatically respond to the variations. In this test, different transmission line impedances are used to investigate the adaptability of the proposed NAPSS. With the change of the transmission line impedance, the extent of the coupling of the controlled generator with the fixed bus can be simulated.

If the transmission line impedance becomes larger, the generator becomes more unstable. A robust controller should be effective for this kind of condition too. In this test, the transmission line impedances of 0.2 *pu* and 0.6 *pu* are used to simulate the tightly and loosely coupled systems. With the power system operating at 0.95 *pu* power, 0.9 *pf* lag, a 0.05 *pu* step decrease in voltage reference at 1 *s* is applied which is removed at 5 *s*. Results are shown in Figs. 5.9 and 5.10.

It is seen that NAPSS can still provide a good response despite the

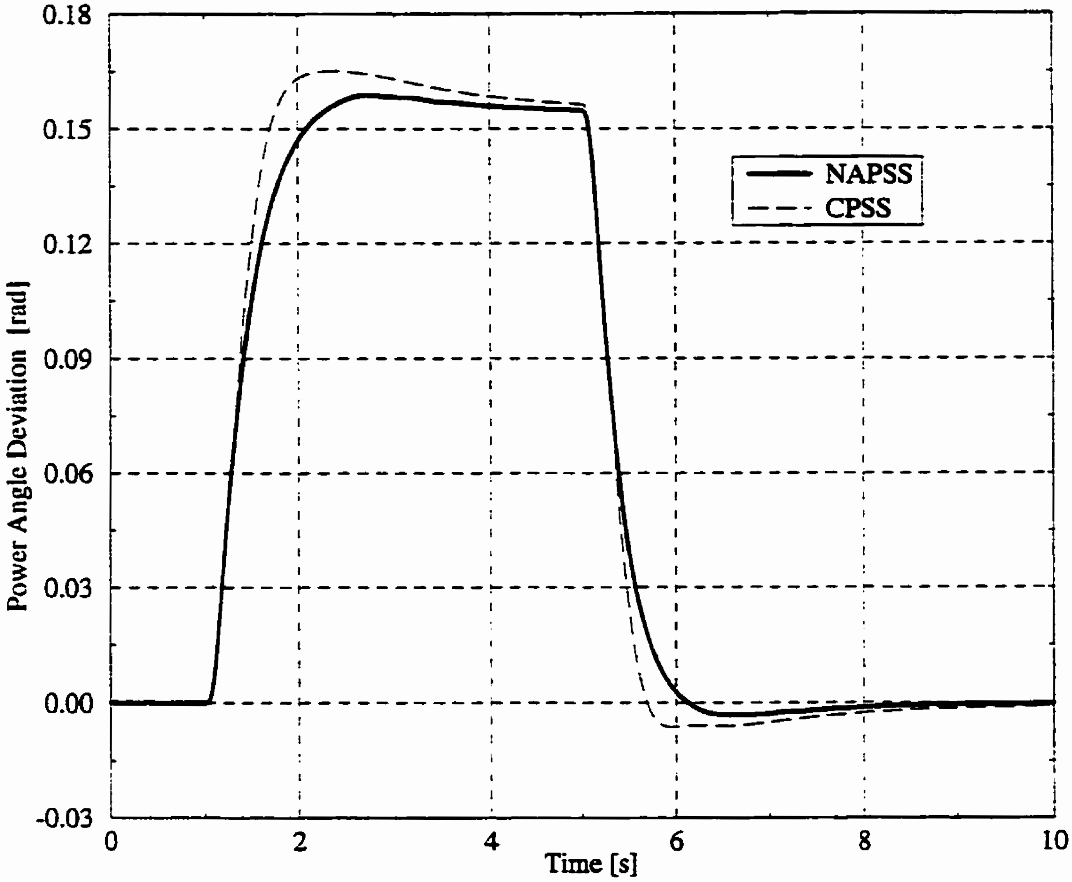


Figure 5.9: Response to a 0.05 *pu* step decrease in voltage reference and return to initial condition with line impedance of 0.2 *pu*.

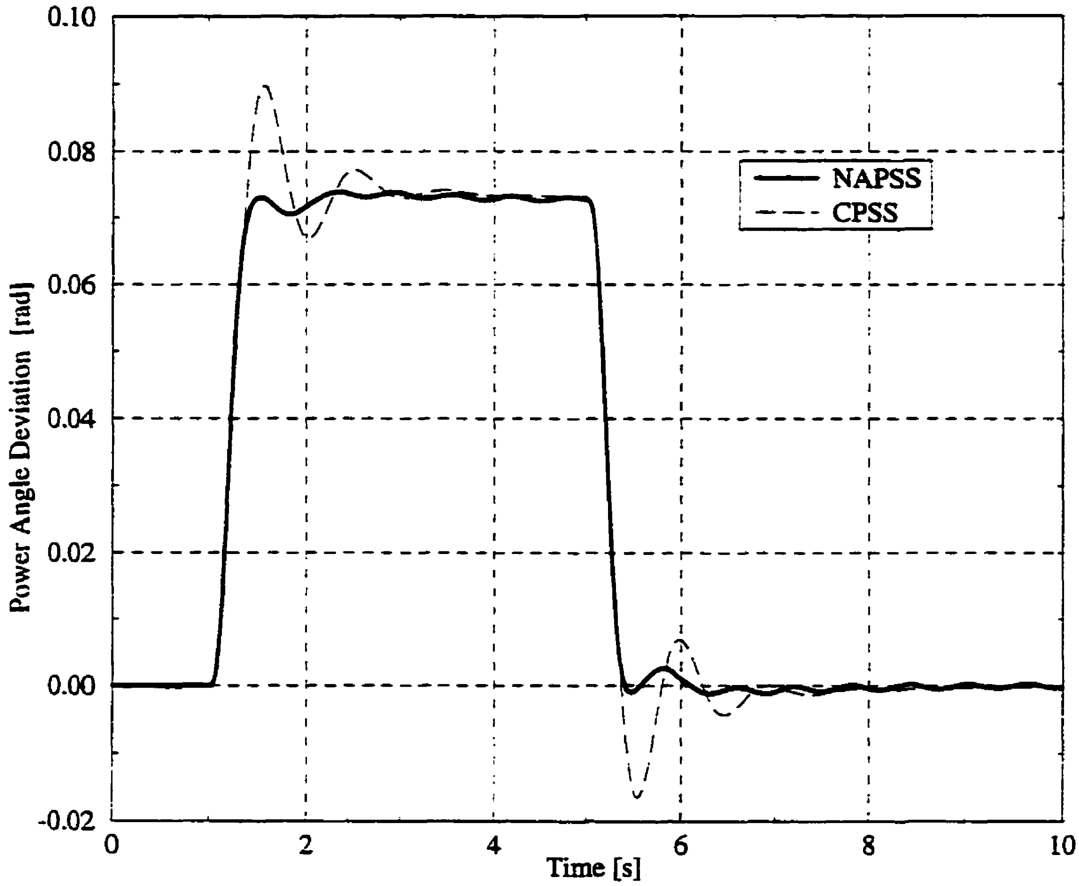


Figure 5.10: Response to a 0.05 *pu* step decrease in voltage reference and return to initial condition with line impedance of 0.6 *pu*.

changes that happen in the system parameters.

5.5.8 Stability Margin

The introduction of the supplementary controller for the power system not only improves the dynamic performance but also increases the stability margin. To demonstrate this fact, a simulation study was conducted. With the initial operating conditions of 0.95 pu power, 0.9 pf lag, the input torque reference was increased gradually. The dynamic stability margin can be described by the maximum power output at which the system loses synchronism. The results for the system without stabilizer, with the CPSS and with the NAPSS are given in Table 5.1. The NAPSS provides the largest output power, which indicates that the dynamic stability margin of the system is improved most by the NAPSS.

Table 5.1: Dynamic stability margin for different stabilizers.

	OPEN	CPSS	NAPSS
Maximum Power	2.65 $p.u.$	3.35 $p.u.$	3.60 $p.u.$
Maximum Rotor Angle	1.55 $rad.$	2.14 $rad.$	2.36 $rad.$

5.6 Summary

Application of the neural adaptive power system stabilizer to single-machine infinite-bus power system is presented in this Chapter. The back-propagation network with on-line learning is used in the proposed stabilizer. The stabilizer introduced here has the following advantages:

- is able to track plant variations;
- considers non-linear nature of the plant;
- does not need states of the plant;
- uses simple (scalar) error vector;
- has a simple structure consisting of 9 neurons in each of the two sub-networks;
- does not require a reference model or teacher signal;
- does not require exact mathematical model of the plant.

Simulation results for various operating conditions and disturbances show that the proposed neural adaptive stabilizer can provide good damping over a wide operating range and significantly and adaptively improves the dynamic performance of the system. The stability margin is also increased by the proposed NAPSS.

Chapter 6

NAPSS Application in Multi-Machine Power System

6.1 Introduction

Simulation studies in Chapter 4 demonstrated that a properly designed NAPSS can provide effective damping of the power system [80], [81]. These studies were on the single-machine infinite-bus environment. The effectiveness of the NAPSS to damp out multi-mode oscillations in multi-machine environment needs to be verified.

The effectiveness of the NAPSS to damp out multi-mode oscillations in a multi-machine environment is investigated in this Chapter. A five-machine power system is used in this study and its transient response to a large disturbance is presented with the multi-mode oscillation phenomenon.

Multi-mode oscillations appear in multi-machine power system in which the interconnected generating units have quite different inertia constants and are weakly connected by transmission lines. These oscillations are generally analyzed in three main oscillation modes, i.e. local, inter-area and inter-machine modes. Depending upon their location in the system, some generators participate in only one oscillation mode, while others participate in more than one mode [37].

6.2 Power System Multi-Mode Oscillations

There are three modes of oscillations in a multi-machine power system:

- *Local Mode*

  Usually refers to oscillations occurring in plant transients stemming from generator rotors oscillating relative to the combined equivalent inertia of the system. This is also described as the generator swinging relative to an infinite-bus formed by the combined equivalent inertia external to a particular generator as shown in Chapter 5. Frequency magnitudes are directly related to the equivalent rotational inertia of the generator and the prime mover, and to the synchronous torque coefficient linking the generator to the fixed bus. Local mode oscillations are in the range of 0.8 to 2 Hz.

- *Inter-Machine Mode*

  This describes frequencies related to closely coupled generators swinging relative to each other. This can occur at a plant that has a diverse mix of generators and controllers or at neighboring plants that are linked with inter-ties such that the machines are relatively closely coupled. Intra-plant frequencies are related to the equivalent machine inertia of the closely coupled generator groups and are in the range of 0.3 to 1 Hz.

- *Inter-Area Mode*

  These frequencies stem from coherent groups of generators in one area swinging relative to a number of other coherent groups in other areas. Inter-area frequencies are in the range of 0.1 to 0.7 Hz and these frequencies may overlap with frequencies described under the other two modes.

6.3 Multi-Machine Power System Model

A five-machine power system without infinite bus, as shown in Fig. 6.1, is used to evaluate the performance of the proposed NAPSS. Five generating units are connected through a transmission network. Generators G_1, G_2 and G_4 have much larger capacities than G_3 and G_5. All five generators are equipped with governors, exciters and AVRs. Parameters of all generators, governors, exciters, AVRs, transmission lines, loads and operating conditions are given in Appendix C. Generators G_2, G_3 and G_5 may be considered to form one area, and generators G_1 and G_4 a second area. The two areas are connected through a tie-line connecting buses 6 and 7. Under normal conditions, each area serves its own load and is almost fully loaded with a small load flow over the tie-line.

Due to the different sizes of the generators and system configuration, multi-mode oscillations occur when the system experiences a disturbance. In order to observe this fact a 0.10 pu step decrease in input torque reference of

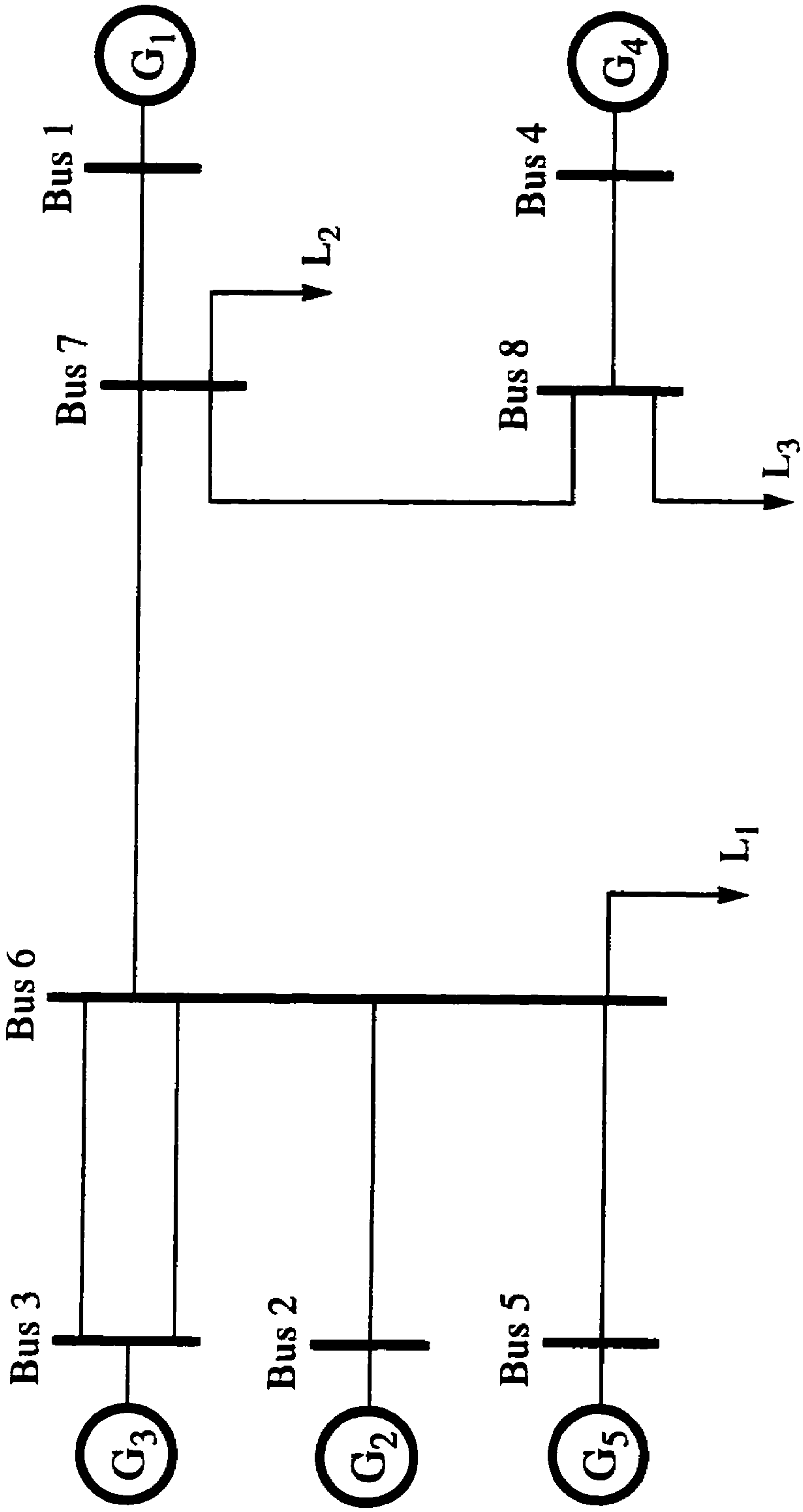


Figure 6.1: Five machine power system.

G_3 is applied at 1 s while the system is operating without any PSSs at the operating point #1 as given in Appendix C. At 10 s, the system returns to its initial condition. Oscillations in Fig. 6.2 show the local mode at about 1.3 Hz and the inter-area mode at about 0.65 Hz. These two frequencies differ significantly due to the large difference in the inertia of the generators. The speed difference between G_2 and G_3 exhibits mainly the local mode oscillations, while the speed difference between G_1 and G_2 shows the inter-area mode oscillations. Both local and inter-area modes of oscillations appear in the speed difference between G_1 and G_3.

6.4 Controller Structure

The structure of the control system for multi-machine study is shown in Fig. 6.3. The input vector to the ANI is

$$[\Delta P_e(k), \Delta P_e(k-1), \Delta P_e(k-2), u(k), u(k-1), u(k-2)] \tag{6.1}$$

where $\Delta P_e(k)$ is the accelerating power and $u(k)$ is the PSS output (generator input), both at time step k. The output of the identifier is the predicted accelerating power, $\Delta \hat{P}_e(k+1)$, at time step $(k+1)$. The input vector to the ANI is scaled before being applied to the network to take a value in the range of $[-1, +1]$. The cost function used for the ANI is

$$J_i(k) = \frac{1}{2}e_i(k)^2 = \frac{1}{2}[\Delta P_e(k) - \Delta \hat{P}_e(k)]^2 \tag{6.2}$$

The weights are updated as

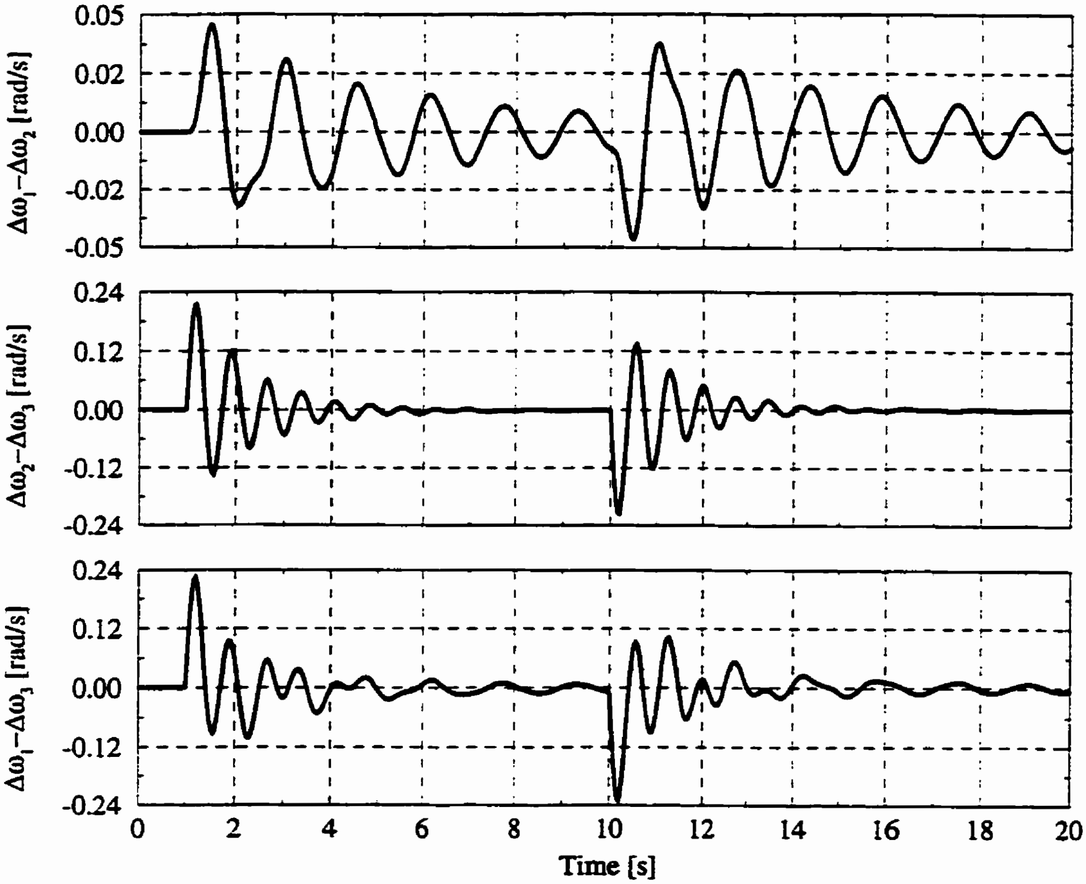


Figure 6.2: Multi-mode oscillations of the five-machine power system.

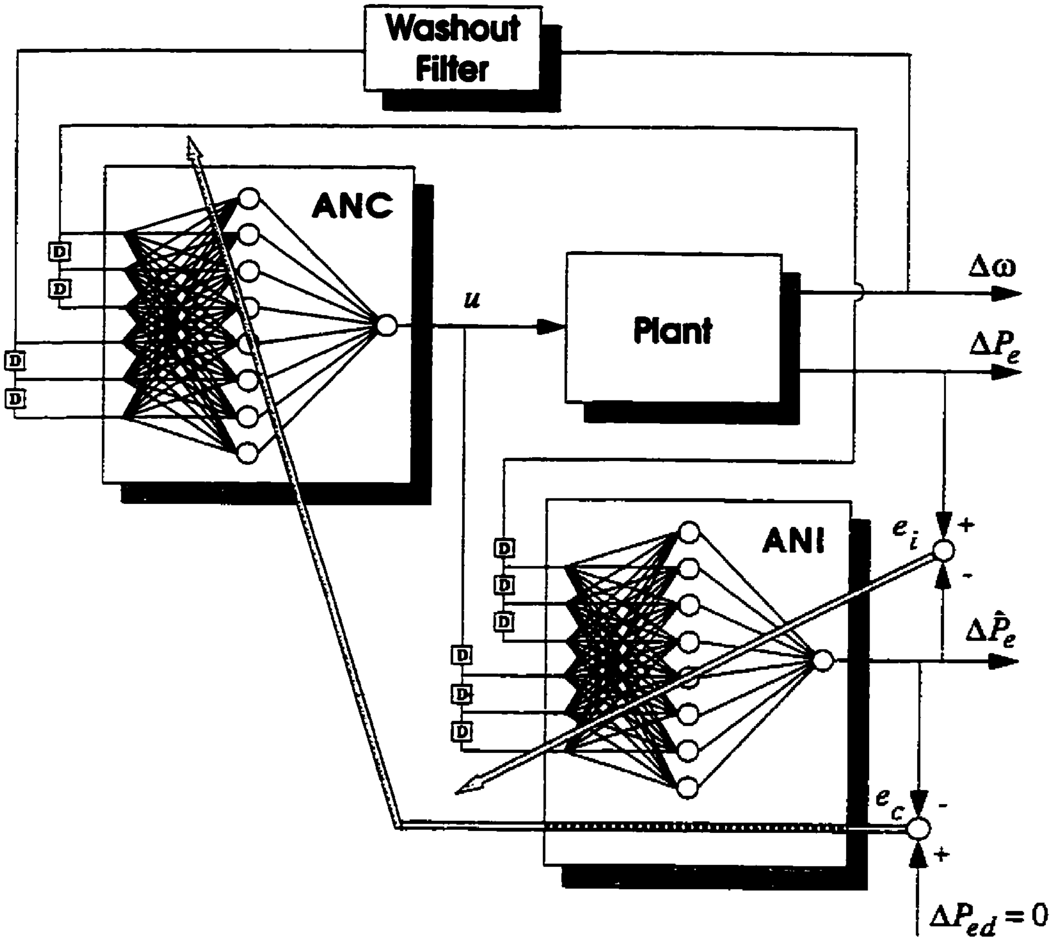


Figure 6.3: Control system structure for multi-machine study.

$$W_i(k) = W_i(k-1) - \eta_i \nabla_{W_i} J_i(k) \tag{6.3}$$

in which $W_i(k)$ is the matrix of identifier weights at time step k and η_i is the learning rate for the ANI. The gradient $\nabla_{W_i} J_i(k)$ is computed by

$$\nabla_{W_i} J_i(k) = -[\Delta P_e(k) - \Delta \hat{P}_e(k)] \frac{\partial \Delta \hat{P}_e(k)}{\partial W_i(k)} \tag{6.4}$$

Using (6.3) and (6.4), the cost function $J_i(k)$ is minimized in each sampling period by back-propagating the scalar error $[\Delta P_e(k) - \Delta \hat{P}_e(k)]$.

The input vector to the ANC is

$$[\Delta P_e(k), \Delta P_e(k-1), \Delta P_e(k-2), \Delta\omega(k), \Delta\omega(k-1), \Delta\omega(k-2)] \tag{6.5}$$

where $\Delta\omega(k)$ is the generator's speed deviation at time step k which goes through a washout filter in order to remove its DC offset. The output of the ANC is the PSS control signal, $u(k)$, at time step k. The inputs to the ANC are also scaled in the range of $[-1, +1]$. The cost function for the ANC is considered as

$$\begin{aligned} J_c(k) &= \frac{1}{2}[e_c(k)^2 + hu(k)^2] \\ &= \frac{1}{2}[\Delta P_{ed}(k) - \Delta \hat{P}_e(k)]^2 + \frac{h}{2}u(k)^2 \end{aligned} \tag{6.6}$$

where $\Delta P_{ed}(k)$, the desired accelerating power at time step k, is equal to zero in a regulatory setup, and h is a tuning parameter used to improve the plant output dynamic characteristics. The weights of the controller, $W_c(k)$, are updated as

$$W_c(k) = W_c(k-1) - \eta_c \nabla_{W_c} J_c(k) \tag{6.7}$$

where η_c is the controller learning rate and the gradient $\nabla_{W_c} J_c(k)$ is defined as

$$\nabla_{W_c} J_c(k) = [\Delta \hat{P}_e(k) \frac{\partial \Delta \hat{P}_e(k)}{\partial u(k)} + h u(k)] \frac{\partial u(k)}{\partial W_c(k)} \tag{6.8}$$

Using (6.7) and (6.8), $J_c(k)$ is minimized each sampling period. As seen in (6.1) and (6.5), the states of the generator are not required for the implementation of the ANI and ANC and only input-output data are used.

6.5 Training of NAPSS

The training of NAPSS has two steps, namely off-line training of the identifier and on-line training of the identifier and controller.

6.5.1 Off-Line Training of Identifier

The success of the NAPSS in suppressing the output oscillations highly depends on the accuracy of the identifier in tracking the plant. This is the very reason to train the ANI off-line before using it in the control algorithm. The training data was gathered with the plant operating over the range of 0.1 *pu* to 1.0 *pu* power output and 0.7 *pf* lead to 0.1 *pf* lag. Disturbances of voltage reference and input torque reference step changes and three phase to ground fault were applied to the system. Using this data, the ANI was trained in the off-line mode employing the back-propagation algorithm [62].

6.5.2 On-Line Training of Identifier and Controller

After off-line training, the ANI is hooked up in the system for further on-line training of both ANI and ANC. The on-line training procedure is composed of the following steps:

1) At time step k, $\Delta\omega(k)$ and $\Delta P_e(k)$ are sampled.

2) Using $\Delta P_e(k)$ and $\Delta \hat{P}_e(k)$, the weights of the ANI are updated, minimizing $J_i(k)$.

3) The output of the controller, $u(k)$, is computed and applied to the generator.

4) Using $u(k)$, the predicted accelerating power, $\Delta \hat{P}_e(k+1)$, is computed by the ANI.

5) Based on $\Delta \hat{P}_e(k+1)$, the weights of the ANC are updated, minimizing $J_c(k)$.

To train the controller based on $\Delta \hat{P}_e(k+1)$, first the weights of the ANI are frozen and the error between the desired and the predicted plant output is back-propagated through the ANI. Then, this back-propagated signal at the input of the ANI is further back-propagated through the ANC to make the required changes to the controller weights.

The errors used to train the ANI and the ANC are both scalar and the learning is done only once in each sampling period for each of the two subnetworks. This simplifies the training algorithm in terms of computation time, which is of special importance in real-time implementation. Also, as it is clear from the above training procedure, the controller is updated based on the output performance and there is no need for a desired controller (external teacher) or a reference model.

6.6 Identifier and Controller Parameters

Both the ANI and the ANC have 6 inputs. There is one hidden layer of 8 neurons with sigmoid nonlinearity and an output layer with one linear neuron, both for the ANI and the ANC. Initial weights of the ANC lie between $[-0.1, +0.1]$, chosen randomly at the beginning of the process. The initial weights of the ANI are set to those obtained from off-line training stage of the ANI as discussed before. The learning rate for the ANI and the ANC is 0.02 and 0.01, respectively. The value of h, the tuning parameter, is set to 2.7. A sampling rate of 25 Hz has been chosen for the digital control system.

6.7 Simulation Results

6.7.1 PSS on One Unit

The proposed NAPSS is first installed on G_3 while none of the other units are equipped with PSS. A 0.10 pu step decrease in input torque reference of G_3 is applied at 1 s which is later removed at 10 s. As shown in Fig. 6.4, the NAPSS damps out the local mode oscillations very effectively. However, as expected, it has little influence on the inter-area mode oscillations. This is because of the fact that the small unit of G_3 does not have enough power to control the inter-area mode oscillations introduced mainly by large units G_1 and G_2. For comparison purposes, a CPSS with the following transfer

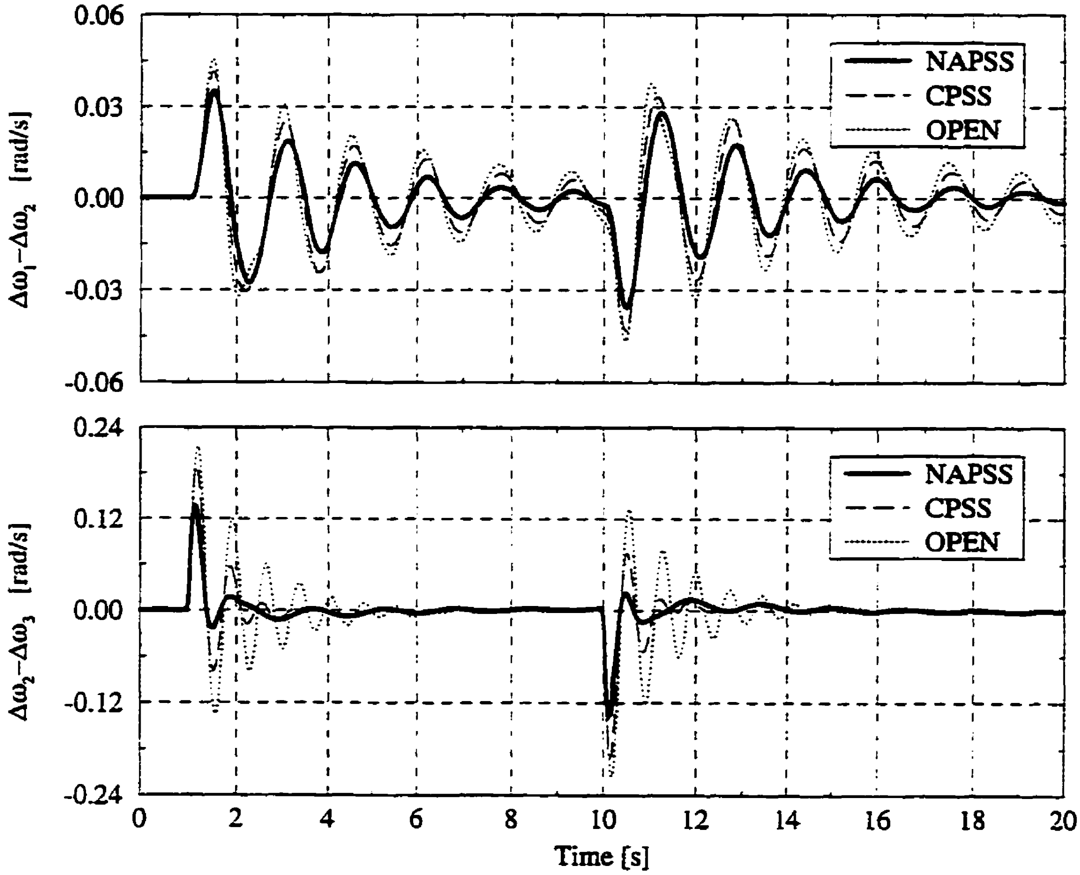


Figure 6.4: System response with NAPSS installed on generator G_3.

function [102] was installed on G_3:

$$U_{PSS}(s) = K_s \frac{sT_5}{1+sT_5} \frac{1+sT_1}{1+sT_2} \frac{1+sT_3}{1+sT_4} \Delta P_E(s) \tag{6.9}$$

After careful parameter tuning, the following parameter set was obtained for the CPSS on G_3.

$$K_s = 1.0,\ T_1 = T_3 = 0.3,\ T_2 = T_4 = 0.1,\ T_5 = 0.4 \tag{6.10}$$

Results of the study with no PSS (OPEN) and with CPSS installed on G_3 are also shown in Fig. 6.4.

6.7.2 PSS on Three Units

To damp out both local and inter-area modes of oscillations, two NAPSSs are additionally installed on G_1 and G_2. Fig. 6.5 shows the response for the same operating conditions and disturbances as before. It can be seen that both modes of oscillations are damped out very effectively. If CPSSs are to be installed additionally on G_1 and G_2 to damp out inter-area mode of oscillations, their parameters have to be re-tuned. After careful parameter tuning, the following parameters are obtained for CPSSs on G_1 and G_2.

$$K_s = 0.3, T_1 = T_3 = 0.07, T_2 = T_4 = 0.03, T_5 = 0.3 \tag{6.11}$$

The responses of the system with no PSS and with CPSSs installed on G_1, G_2 and G_3 are shown in Fig. 6.5.

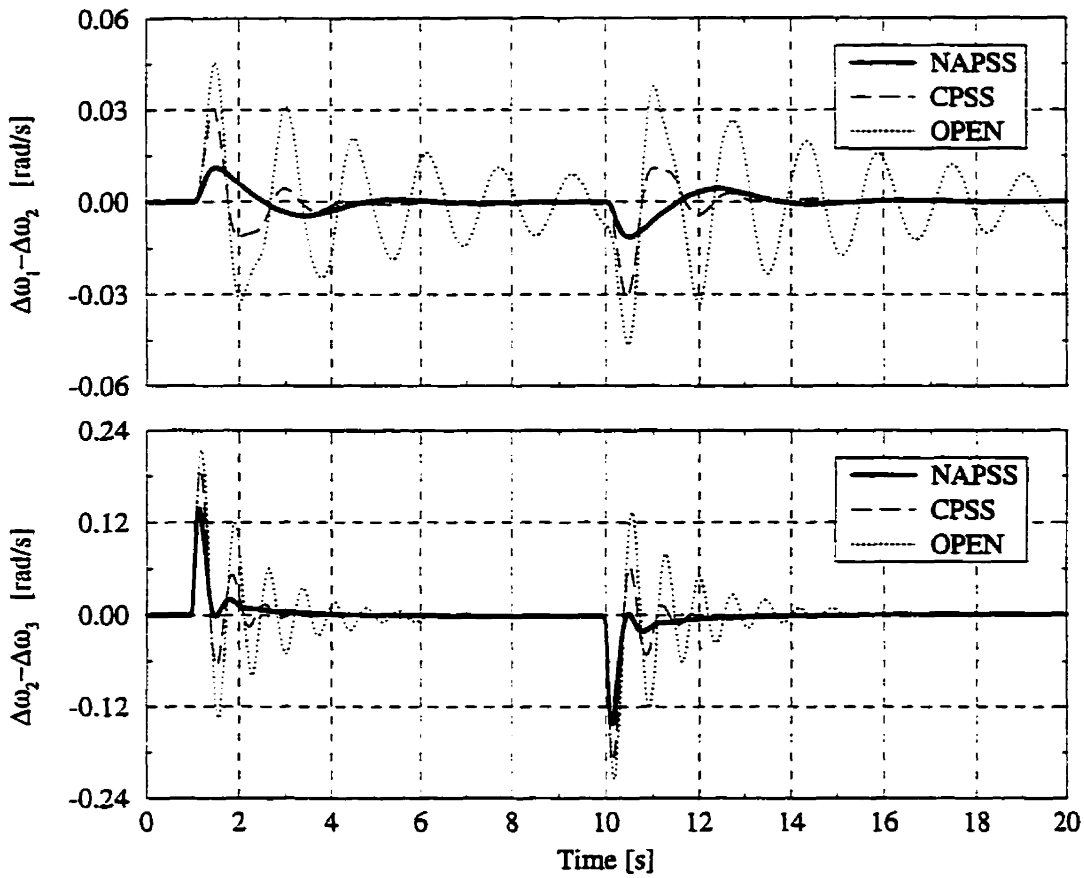


Figure 6.5: System response with PSSs installed on G_1, G_2 and G_3.

6.7.3 Three-Phase to Ground Fault Test

Having the same operating condition as before, a three-phase to ground fault is applied at the middle of one transmission line between buses 3 and 6 at 1 s and cleared 100 ms later. At 10 s, the faulted line is restored successfully. This disturbance is large enough to cause the system to operate in the non-linear region. Fig. 6.6 shows the response of the system with no PSS, NAPSSs only and CPSSs only installed on G_1, G_2 and G_3. It is seen that the NAPSS improves the system response. This is due to the adaptive property of the NAPSS.

6.7.4 Self-Coordination Ability of NAPSS

One of the important features of the NAPSS is its self-coordinating property. The NAPSS can coordinate itself with the existing PSSs in the system automatically due to its on-line learning property. To demonstrate this fact, the NAPSS is installed on G_1 and G_3 and CPSS with proper parameter set on G_2, G_4 and G_5. Fig. 6.7 shows the response for 0.15 pu step decrease in torque reference of G_3 at 1 s and return to initial condition at 10 s. It can be seen that all PSSs work cooperatively to achieve a good performance.

6.7.5 New Operating Condition Test

To test the performance of the NAPSS under other operating conditions, the operating point of the system is set to operating point #2 as given in Appendix C. Fig. 6.8 shows the response under a three-phase to ground fault having no PSS (OPEN), CPSS and NAPSS on G_1, G_2 and G_3. It is shown that NAPSSs can damp out the oscillations very effectively even as

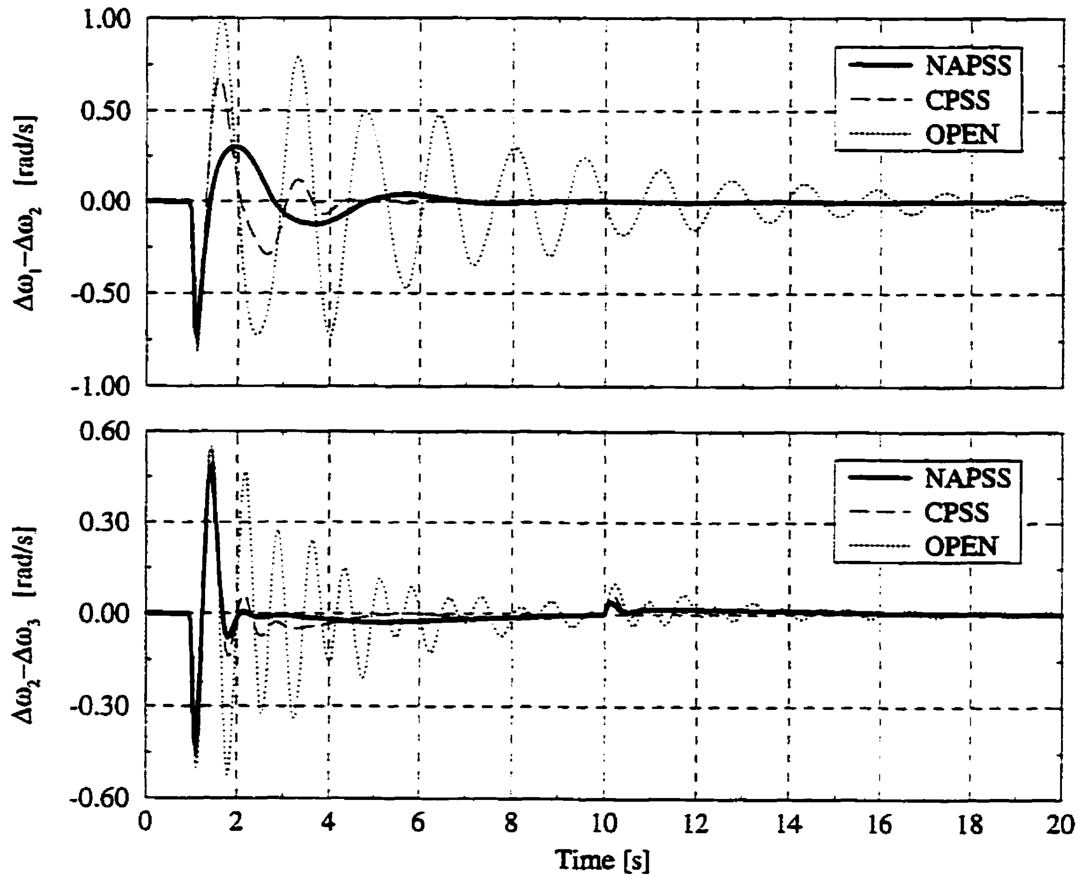


Figure 6.6: System response to a three phase to ground fault with PSSs installed on generators G_1, G_2 and G_3.

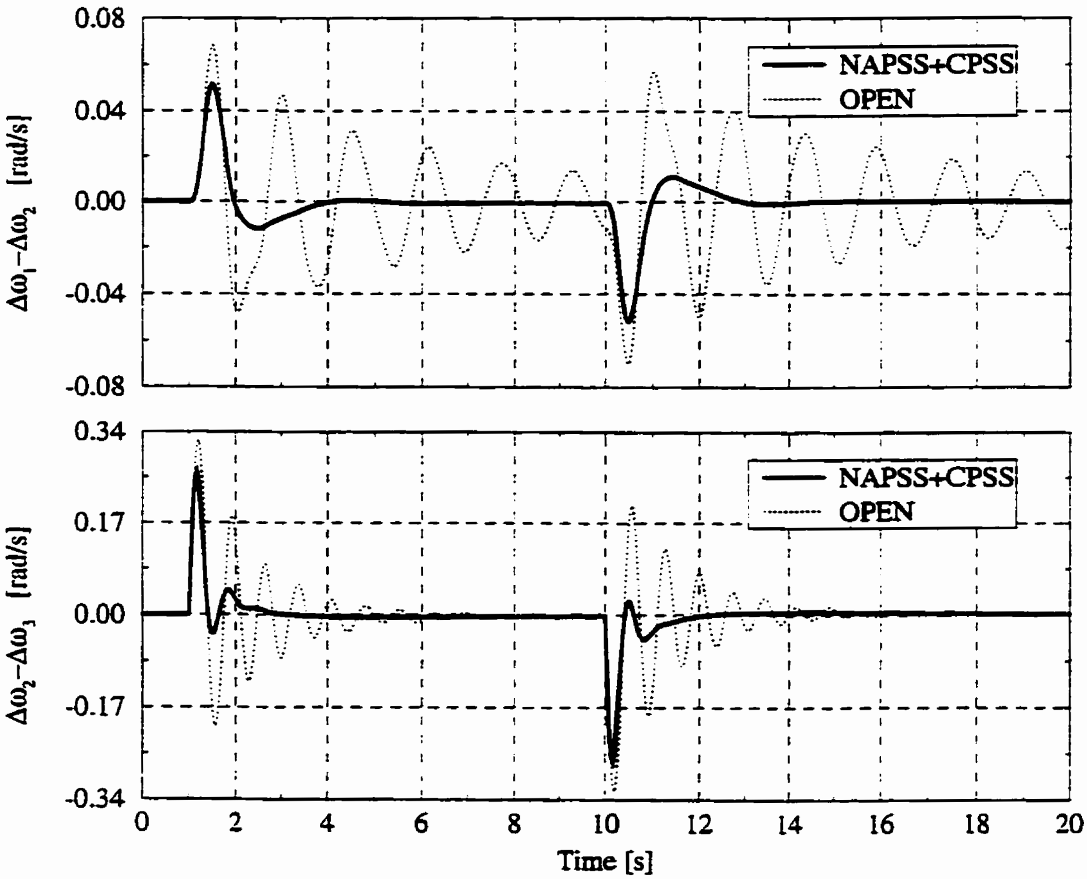


Figure 6.7: System response with NAPSS installed on generators G_1 and G_3 and CPSS on G_2, G_4 and G_5.

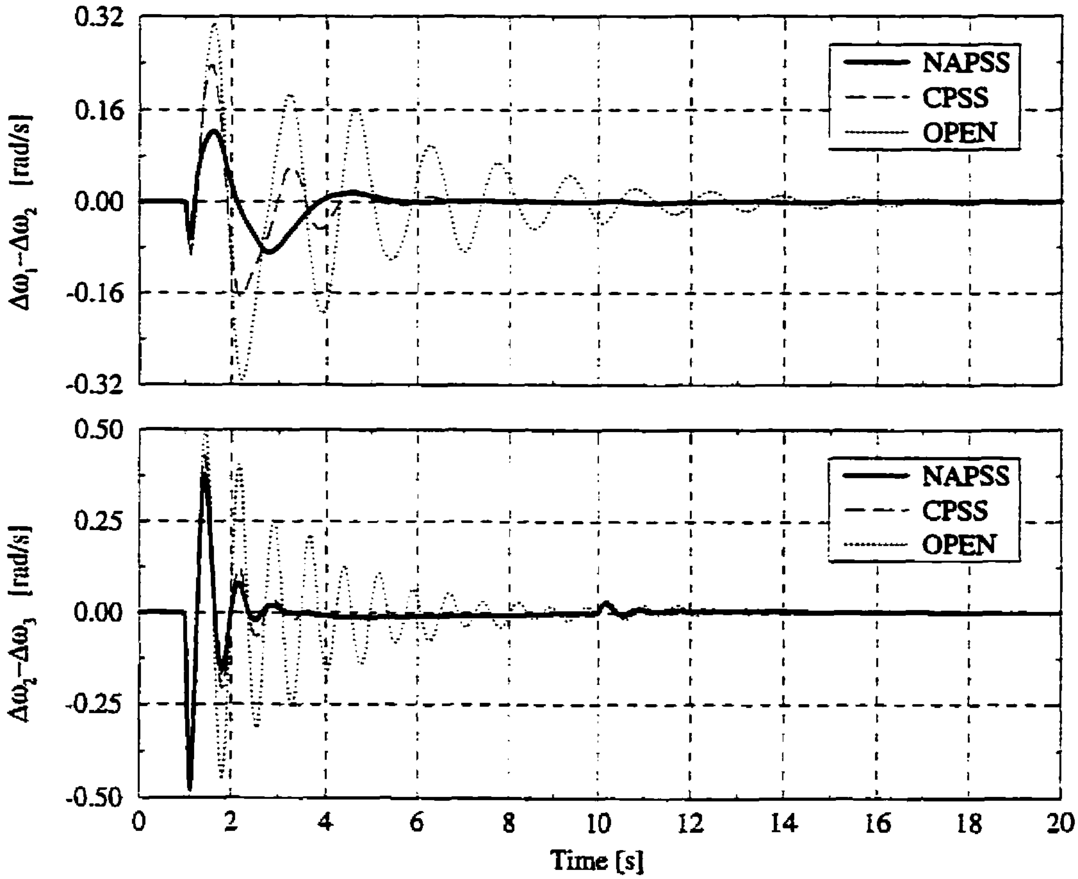


Figure 6.8: System response to a three phase to ground fault with PSSs installed on generators G_1, G_2 and G_3 for the new operating point.

the operating conditions change.

6.8 Summary

Multi-mode oscillations appear in a multi-machine power system in which the interconnected generating units have quite different inertias and they are weakly connected by transmission lines. Application of a neural adaptive power system stabilizer (NAPSS) to a five-machine power system is described in this chapter. The proposed NAPSS employs back-propagation network with on-line learning. Its structure is very simple and there is no need for a large number of neurons in its implementation. The accelerating power and speed deviation of the unit are used as inputs to the NAPSS. The stabilizer is trained in each sampling period by input-output data using a simple single-element error vector. Due to its adaptability, the NAPSS can adjust itself to different conditions to effectively damp out both local and inter-area modes of oscillations. The self-coordinating ability of NAPSS is also demonstrated.

Part III

Experimental Tests

Chapter 7

NAPSS Laboratory Implementation and Real-Time Test Results

7.1 Introduction

Results in the previous chapters have shown that NAPSS exhibits very good control performance [80], [103]. Like most other neural network based controller research, the results in the previous chapters are based on computer simulations. In these simulations, the power system was simulated by using a set of simultaneous differential equations, and NNs were simulated by using a sequential algorithm to simulate the parallel distributed nature of NNs. Since the power system simulation models represent fairly closely the physical system and the sequential algorithm can get the same output except for a longer computation time, the simulation results presented in the previous chapters may be said to be close to those expected in a physical system.

Computer simulation is different from the real physical system since the operating environment of the physical system is not ideal and there exist noise and saturation of elements as well as unexpected disturbances that cause the power system to operate under a continuous small perturbation. Therefore, after theoretical development and computer simulation, the next desirable step is to evaluate the control strategy on a physical model of the controlled system.

In general, it is necessary to implement a controller in hardware. Thus, the design of the NAPSS is not finished until the hardware implementation is finished. There are some practical considerations that need to be looked at in NAPSS implementation in the laboratory. These are:

- what type of physical plant can be used in the laboratory;
- in what environment the NAPSS can be built;
- how to implement the parallel distributed nature of NN.

Because of the above considerations, very few laboratory implementations of the neural network based controllers are reported in the literature. An early stage laboratory implementation of an NN based controller to simulate an adaptive trajectory controller for a DC motor is described in [104]. In this work, different NNs were trained to simulate the identification part and control part of the adaptive controller. An off-line trained neural network based controller was also implemented in [105].

So far as is known, the first implementation of an on-line trained neural network based PSS in a laboratory environment is reported in this chapter. A laboratory physical model of a power system, which has the same characteristics as that of a real power system, is set up. Since the neural network hardware capable of handling on-line training was not available at the time of implementation, a sequential simulation method was employed to implement the NAPSS in software on a Digital Signal Processor (DSP) board mounted on a 80386 PC. For comparison, a digital conventional PSS (CPSS) is implemented in the same environment on the DSP board. Details of implementation along with the experimental results are described in this chapter. Effectiveness of the NAPSS in response to various types of disturbances for a variety of operating conditions is demonstrated.

7.2 Power System Physical Model

Schematic diagram of the physical model of a single-machine infinite-bus power system available in the Power System Research Laboratory at the University of Calgary is shown in Fig. 7.1. It consists of a 3 phase, 3 kVA, 220 V synchronous micro-alternator driven by a $220V$, 30 A DC motor. The alternator is connected to the city power system (constant voltage bus) through two parallel transmission lines. The parameters of the physical system are given in Appendix D. The lumped element physical model of the transmission line simulates the performance of a 500 kV, 300 km long double circuit transmission line which consists of six π-sections. It has a frequency response which is close to that of an actual transmission line up to 500 Hz. A Time Constant Regulator (TCR) is used to change effective field time constant of the generator in order to emulate a large generating unit. Using this circuit, effective field time constant of the generator can be increased up to

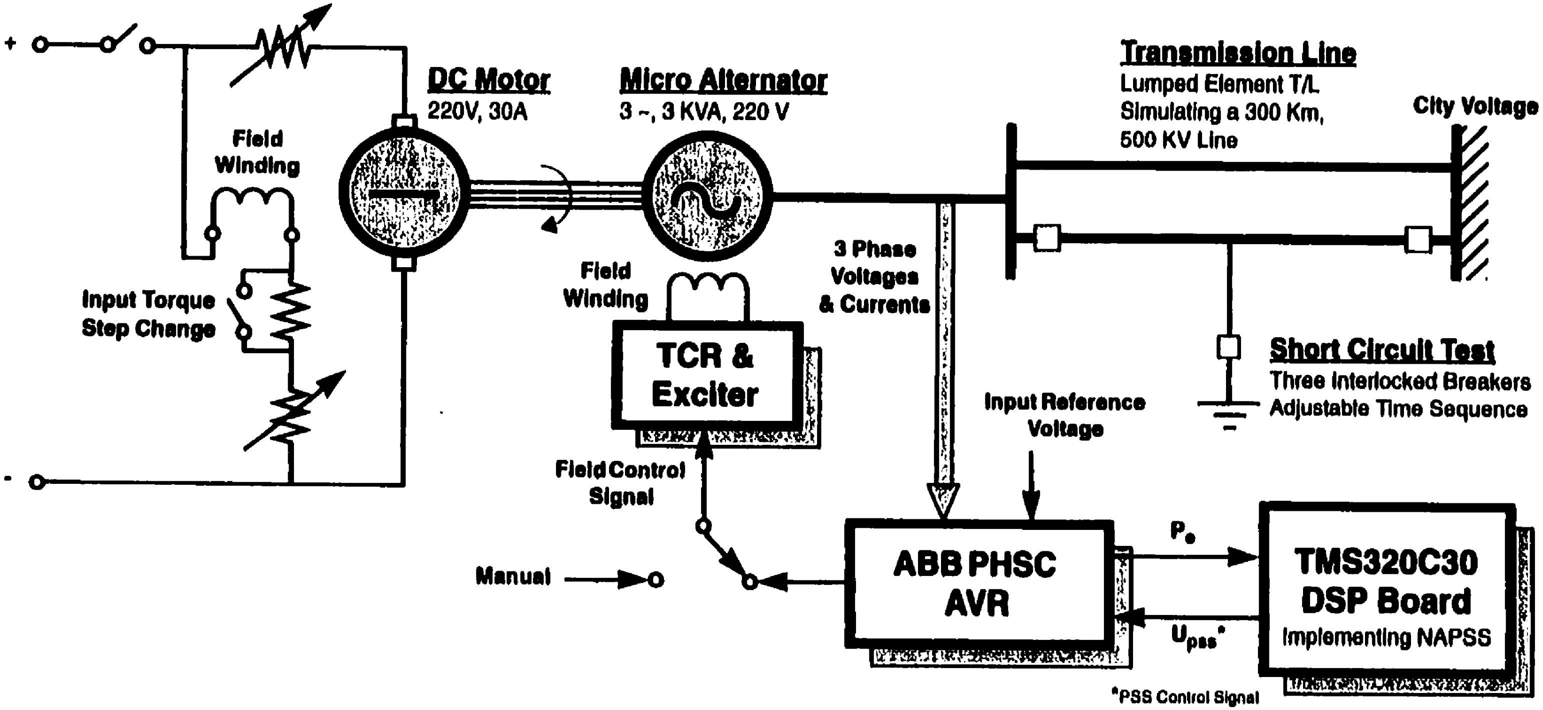


Figure 7.1: Structure of the power system physical model.

10 s.

The system is also equipped with a commercial ABB PHSC AVR implemented on a Programmable Logic Controller (PLC) to control the terminal voltage of the generating unit. It is programmed using a function block programming language called FUPLA. Three phase AC voltages and currents at the generator terminal are stepped down, rectified and filtered with a cut-off frequency of 8 Hz to form six DC input signals to the AVR. The PLC-based AVR computes the required field control signal which is fed to the TCR. The AVR also calculates the active power signal which is used as the PSS input. The NAPSS is implemented on a DSP board based on the TMS320C30 DSP chip. It computes the required control signal, U_{pss}, to be fed to the AVR. Details of the DSP hardware and its connection to the AVR are covered in the next section.

A variety of disturbances can be applied to the system. Using the switch shown in the excitation circuit of the DC motor, Fig. 7.1, a step change in input torque of the generator can be applied. Similarly, the input reference voltage of the AVR can be stepped down or up. In addition, different types of faults can be applied to simulate large disturbances. The operating condition of the generator, i.e. active power and power factor, can also be changed by changing the armature current of DC motor and terminal voltage of the generator respectively.

7.3 Implementation of NAPSS

7.3.1 Sequential Implementation of Parallel Mechanism

Parallel processing is one of the most important properties of the neural networks. The neurons in a layer operate in parallel. This results in high speed operation. Research on designing neural networks by using VLSI technology is advancing very fast. However, because NN hardware with on-line learning capability was not available in the laboratory at the time of implementation, a sequential implementation method was designed to simulate the parallel mechanism.

The sequential implementation is very similar to the NN simulation used in the simulation studies in the previous chapters. Instead of allowing all neurons in the same layer to compute simultaneously, this method only allows neurons to compute one after another. The computation starts from the first neuron of the first layer and ends with the last neuron of the output layer. The output of each neuron is held constant until the next computation cycle starts. This method can get the same output as that of a real neural network chip except that the computation time is much longer here.

7.3.2 Hardware Structure

Structure of the digital control system is shown in Fig. 7.2. The NAPSS is developed on a DSP board supplied by SPECTRUM Signal Processing Inc. It contains a Texas Instruments TMS320C30 DSP chip. The chip is a 32-bit floating point device with a speed of 16.7 million instructions per second. Its performance is further enhanced through its large on-chip memories,

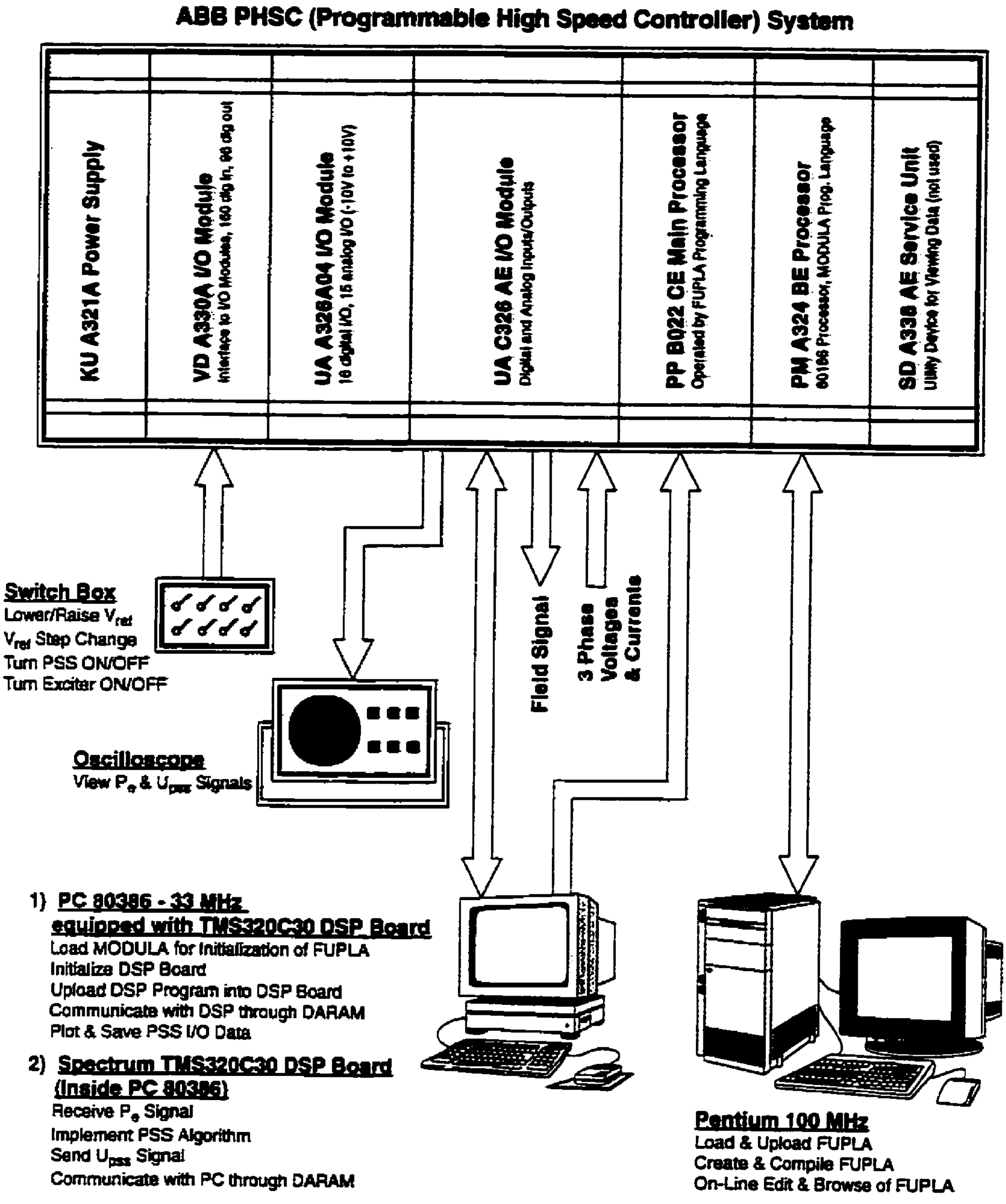


Figure 7.2: Digital control system structure.

concurrent DMA controller, two external interface ports and an instruction cache. Two 200 kHz, 16-bit analog I/O channels on board, coupled with direct access to all serial and parallel I/O channels of DSP chip, provide the exterior input-output functions. The 32-bit on-chip timer is programmed by software to a resolution of 120 ns. The board is mounted inside a PC which is equipped with corresponding development and debugging tools.

The AVR calculates the generator active power, P_e, based on the measured instantaneous voltages and currents. The P_e signal is then transfered to DSP board through the A/D channel. This A/D channel samples the signal at 200 Hz. The sampled signal goes through a filter, which limits the noise and provides anti-aliasing protection. The filtered signal is then stored in a buffer. The DSP chip reads the buffer and computes the control signal, U_{pss}. The computed U_{pss} is fed to the D/A channel which filters the signal for smoothing before sending it out. The AVR receives the PSS control signal as a supplementary input and adds it to the voltage reference signal. The combined signal then goes through the AVR block in order to make the required field control signal to the TCR.

7.3.3 Software Structure

The NAPSS software, running on DSP, is developed in C and Assembly languages. In addition, a Man-Machine-Interface (MMI) routine, running on PC, is also developed to further improve the PSS development and implementation environment. Flow chart of the MMI routine is shown in Fig. 7.3. This routine functions as a supervisor monitor. It first initializes I/O vectors for DSP-PC communication. Then, it loads the DSP code into the

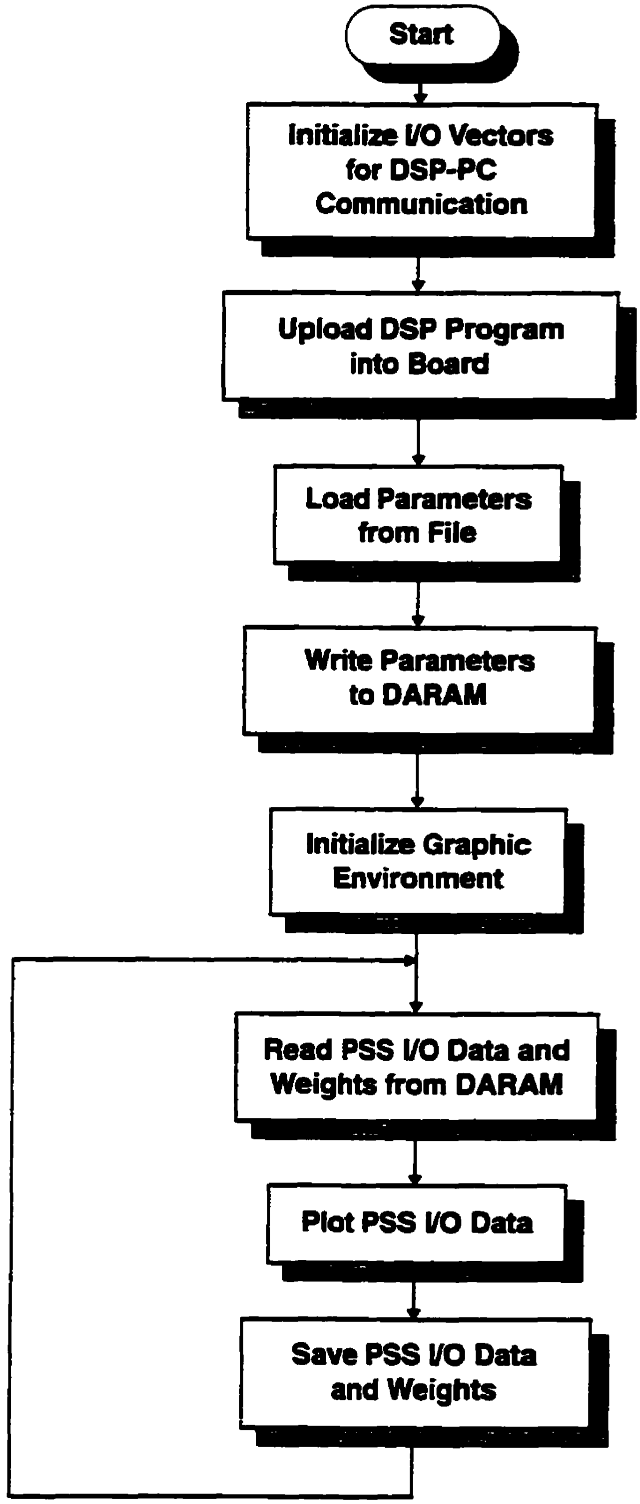


Figure 7.3: Flow chart of the Man-Machine-Interface (MMI) program.

chip. It then reads control parameters and sends them to DSP chip through Dual Access RAM (DARAM). After the inception of main control loop by DSP, the MMI routine reads the input-output data of the controller running on the DSP board every 50 ms. These data are plotted on-line on the screen, and also can be forwarded to a file for further analysis.

Flow chart of the program running on the DSP board is shown in Fig. 7.4. This program first initializes the I/O vectors for DSP-PC communication, then reads control parameters from DARAM. After initializing A/D and D/A channels and sampling time counter, it enters the main control loop. There, it iteratively reads the input signal, processes that signal, computes the controller output and sends the output signal along with the neural network weights to DARAM.

7.4 Control Strategy

A schematic diagram of the controller architecture is shown in Fig. 7.5.

The input vector to the ANI is:

$$[\Delta P_e(k), \Delta P_e(k-1), \Delta P_e(k-2), u(k), u(k-1), u(k-2)] \tag{7.1}$$

where $\Delta P_e(k)$ is the active power deviation and $u(k)$ is the PSS output (generator input), both at time step k. The active power deviation is obtained by removing the DC offset of the generator active power using a washout filter. The output of the identifier is the predicted active power deviation,

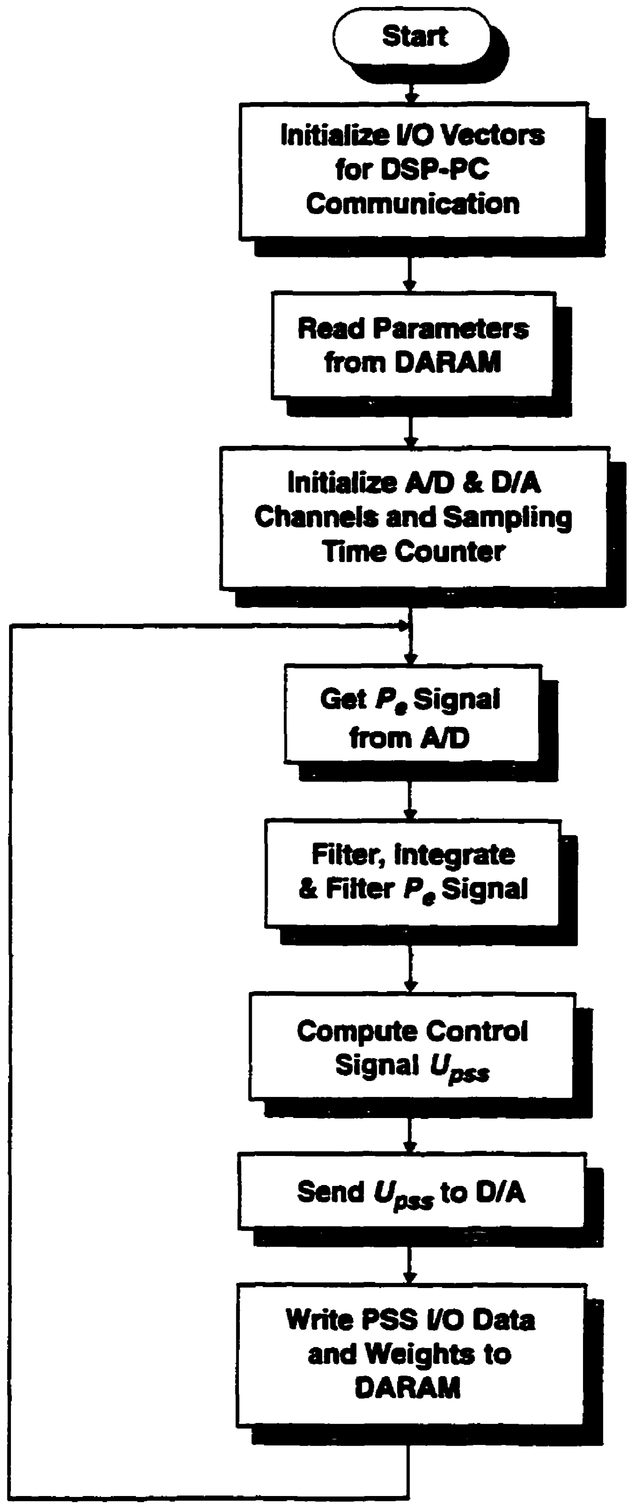


Figure 7.4: Flow chart of the DSP program.

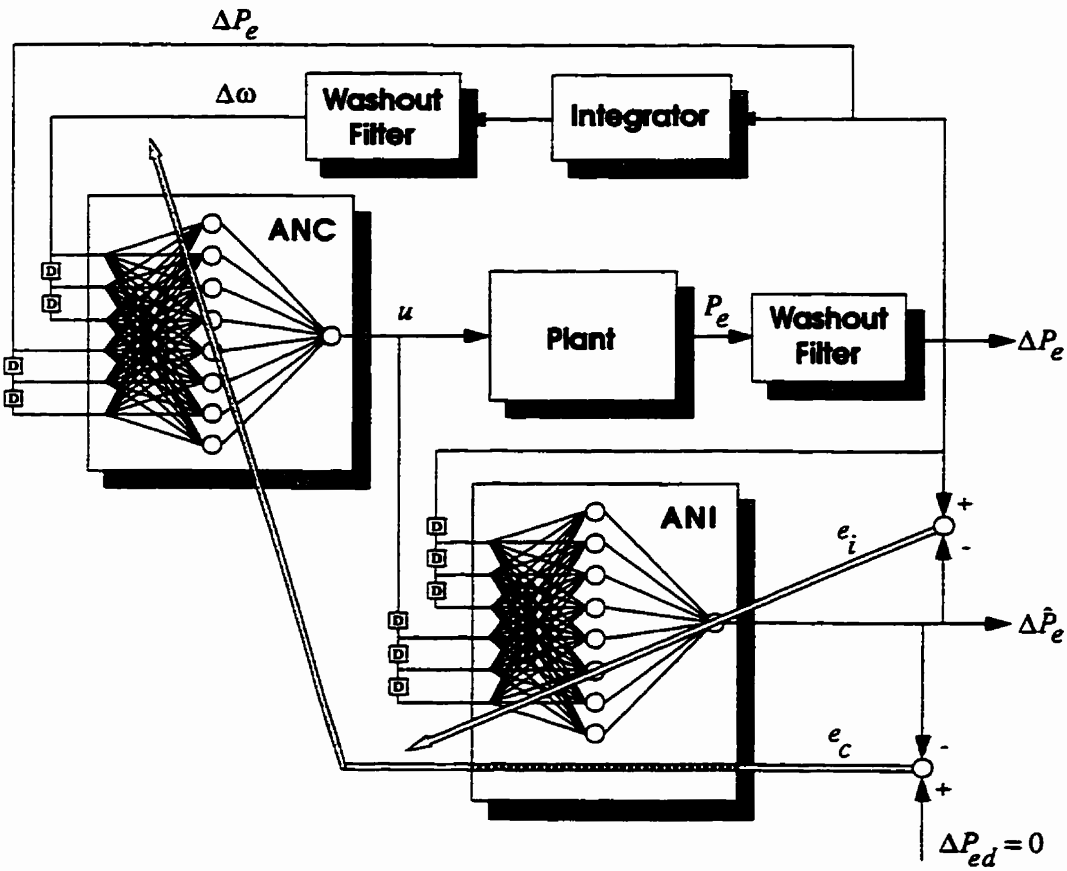


Figure 7.5: Control system architecture in implementation stage.

$\Delta\hat{P}_e(k+1)$, at time step $(k+1)$. The input vector to the ANC is

$$[\Delta P_e(k), \Delta P_e(k-1), \Delta P_e(k-2), \Delta\omega(k), \Delta\omega(k-1), \Delta\omega(k-2)] \quad (7.2)$$

where $\Delta\omega(k)$ is a signal proportional to the generator speed deviation. This signal is obtained by integrating the ΔP_e signal. Again, a washout filter is used to remove the DC offset of this signal.

7.5 Training Procedure

The on-line training procedure is composed of the following steps:

1) At time step k, $P_e(k)$ is sampled and $\Delta P_e(k)$ and $\Delta\omega(k)$ are computed.

2) Using $\Delta P_e(k)$ and $\Delta\hat{P}_e(k)$, the ANI is trained.

3) The output of the controller, $u(k)$, is computed.

4) Using $u(k)$, the predicted active power deviation, $\Delta\hat{P}_e(k+1)$, is computed by the ANI.

5) Based on $\Delta\hat{P}_e(k+1)$, the ANC is trained.

To train the ANC, first the weights of the ANI are frozen and the error between the desired and the predicted plant output is back-propagated through the ANI. This back-propagated signal at the input of the ANI is further back-propagated through the ANC to make the required changes to the controller weights.

Since the sampling period can not be less than the time needed for back-propagation of error and updating the weights of the neural network, special care should be given to the selection of the network size and error vector size. If the network is big and/or the error vector is large, the on-line training algorithm can not be accomplished in one sampling period. Here, the errors used to train the ANI and the ANC are both scalar and the learning is done only once in each sampling period for each of the two subnetworks. This simplifies the training algorithm in terms of computation time. Also, the controller is updated based on the output performance and there is no need for a desired controller (external teacher) or a reference model.

Both the ANI and the ANC have a 6x8x1 structure, based on earlier experience (see Chapter 5) [80]. This leads to a simple network with a total of 9 neurons in each of the two subnetworks. Having a simple network is very important in real-time implementation, since it involves less computation which allows a small sampling time. This in turn results in a better performance. The 8 neurons of hidden layer have sigmoid non-linearity and the single output neuron is a linear one. The learning rate for the ANI and the ANC is 0.01 and 0.03 respectively.

Using computer simulation and parameters of the system given in Appendix D, both the ANI and ANC are first trained on a SUN Sparc Station platform employing the on-line version of the back-propagation algorithm [62]. After this stage, the weights of the neural networks are plugged into the real system for further on-line training.

7.6 Implementation of CPSS

A CPSS is digitally implemented on the same digital control environment for comparison purposes. Having the analog transfer function of:

$$U_{pss}(s) = K_s \frac{sT_5}{1+sT_5} \frac{1+sT_1}{1+sT_2} \frac{1+sT_3}{1+sT_4} \Delta P_E(s) \tag{7.3}$$

the CPSS is discretized using the bilinear transformation, $s = \frac{2}{\tau}\frac{z-1}{z+1}$, where τ is the sampling period. Since the washout filter is implemented in another block, only the lead-lag element needs to be discretized. After applying this transformation, the digital CPSS would have the following transfer function:

$$u(k) = \frac{g'_0 + g'_1 z^{-1} + g'_2 z^{-2}}{f'_0 + f'_1 z^{-1} + f'_2 z^{-2}} \Delta P_e(k) \tag{7.4}$$

where coefficients $\{g'_i\}$ and $\{f'_i\}$ are explicit function of gain K_s and time constants $T_1 - T_4$. The sampling period for the digital CPSS is chosen to be $\tau = 1\ ms$.

7.7 Experimental Results

The performance of the proposed NAPSS has been investigated by a number of experimental tests for a variety of operating conditions and disturbances. For the sake of brevity, however, results of only a representative set of studies are presented here. For comparison purposes, results of the same tests using digital CPSS are also included. All experimental data are collected by the MMI routine and saved automatically for further analysis. However, in order to make the disturbances seem to happen at the desired time point, the time axis is adjusted.

The inherent damping of the physical model power system used for experimental tests is quite high. However, to evaluate the supplementary damping effect provided by PSS, a small system damping is desired. To achieve this goal, only one transmission line is in operation for all the tests except the three-phase to ground fault test. The sampling time for the digital control system is 5 ms.

7.7.1 Voltage Reference Step Change

With the generating unit operating at 0.9 pu power, 0.85 $pf\ lag$ and terminal voltage of 1.1 pu, a 0.10 pu step increase in voltage reference is applied at 1 s. At time 8 s, the change in input reference voltage is removed and the system returns to its original operating condition. The generator active power deviation with NAPSS and without PSS (OPEN) are shown in Fig. 7.6. For the open-loop system, when the voltage reference drops, the oscillations become more severe. This is because the stability margin is reduced when the voltage drops. Therefore, in Fig. 7.6, the generator is in a more stable situation at 1 s compared to its situation at 8 s.

In order to make a comparison between CPSS and NAPSS, the parameters of the CPSS are carefully tuned to give the best response for the operating conditions of this test. These parameters are given in Appendix D. The response of the system with the CPSS having these parameters is also given in Fig. 7.6. It is obvious from the figure that both NAPSS and CPSS are producing good results.

To further test the performance of the NAPSS, the operating condition

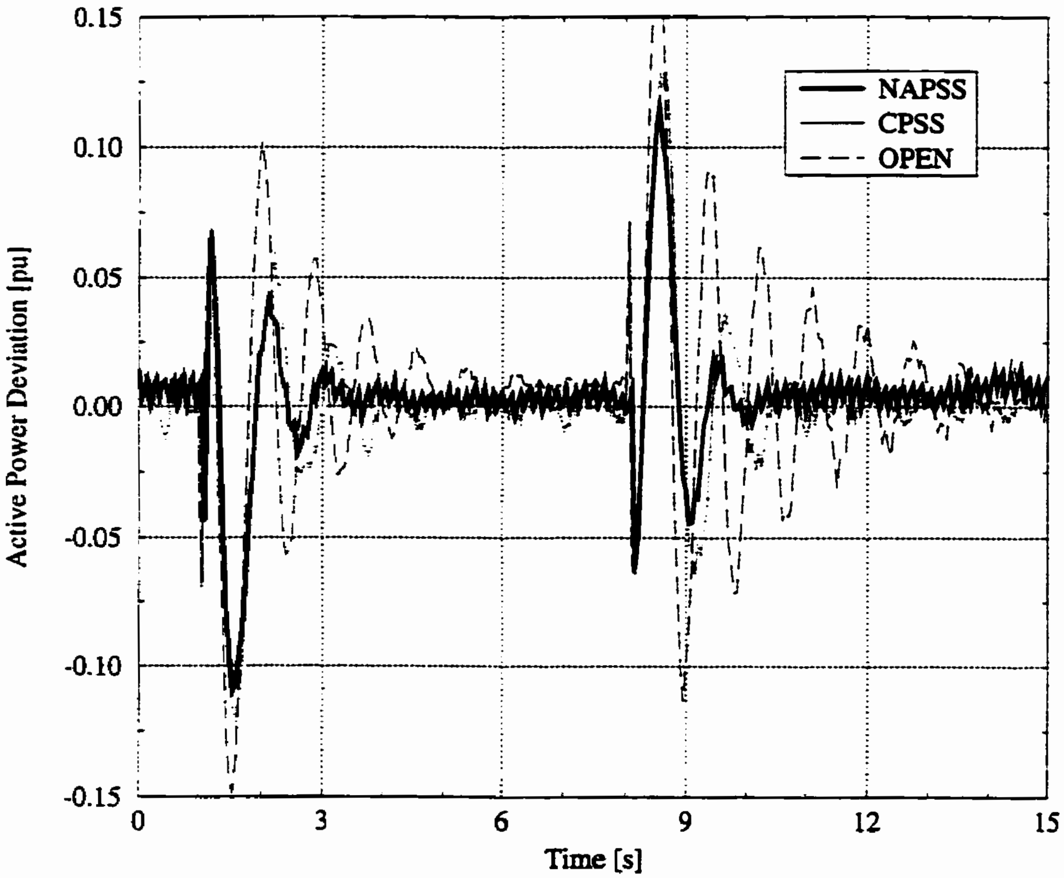


Figure 7.6: System response to a 0.10 *pu* step disturbance in voltage reference, P=0.9 *pu* and pf=0.85 *lag*.

is changed to 0.96 *pu* power, 0.96 *pf lead* and 1.0 *pu* terminal voltage. The same disturbance of 0.10 *pu* step change in input reference voltage is applied with the same timing. System responses to this disturbance having NAPSS, CPSS and no PSS are shown in Fig. 7.7. Although the stability margin is reduced in the new operating condition, the NAPSS still provides a good performance.

7.7.2 Input Torque Reference Step Change

In this test, the generator is operating at 0.9 *pu* power, 0.85 *pf lag* and 1.1 *pu* terminal voltage. A 0.25 *pu* step decrease in input torque reference is applied at 1 *s* and removed at 8 *s*. The system response is given in Fig. 7.8. For this new operating point and disturbance, the NAPSS still provides a quick and well-damped response.

Another disturbance of 0.22 *pu* step decrease in input torque reference is applied to the system while the generator is operating in the leading power factor condition of 0.82 *pu* power, 0.96 *pf lead* and 1.0 *pu* terminal voltage. The response for this test is shown in Fig. 7.9. The figure clearly demonstrates the effectiveness of the NAPSS. The control signals of both NAPSS and CPSS are given in Fig. 7.10.

7.7.3 Three-Phase to Ground Fault Test

Even though the PSS is not specially designed for the purpose of improving stability under transient conditions, it does exert a positive influence during the recovery period after the disturbance.

In order to evaluate the performance of the NAPSS under transient con-

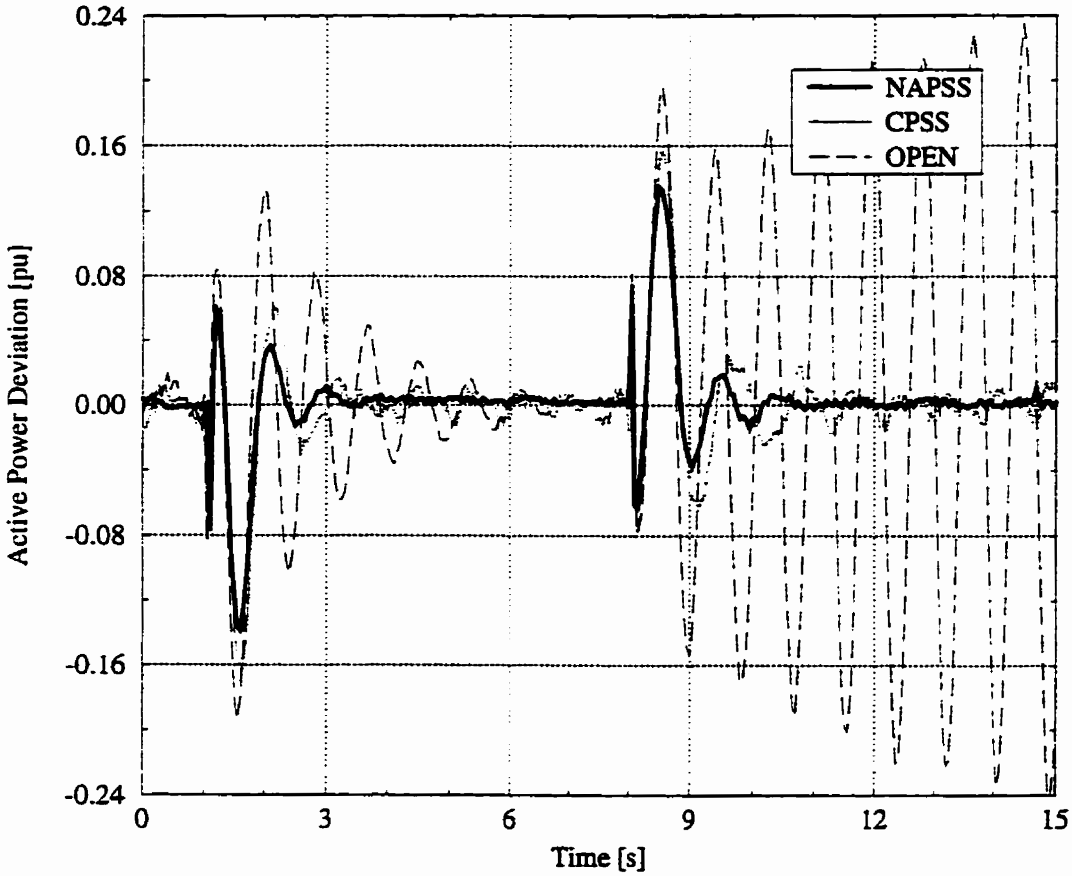


Figure 7.7: System response to a 0.10 *pu* step disturbance in voltage reference, P=0.96 *pu* and pf=0.96 *lead*.

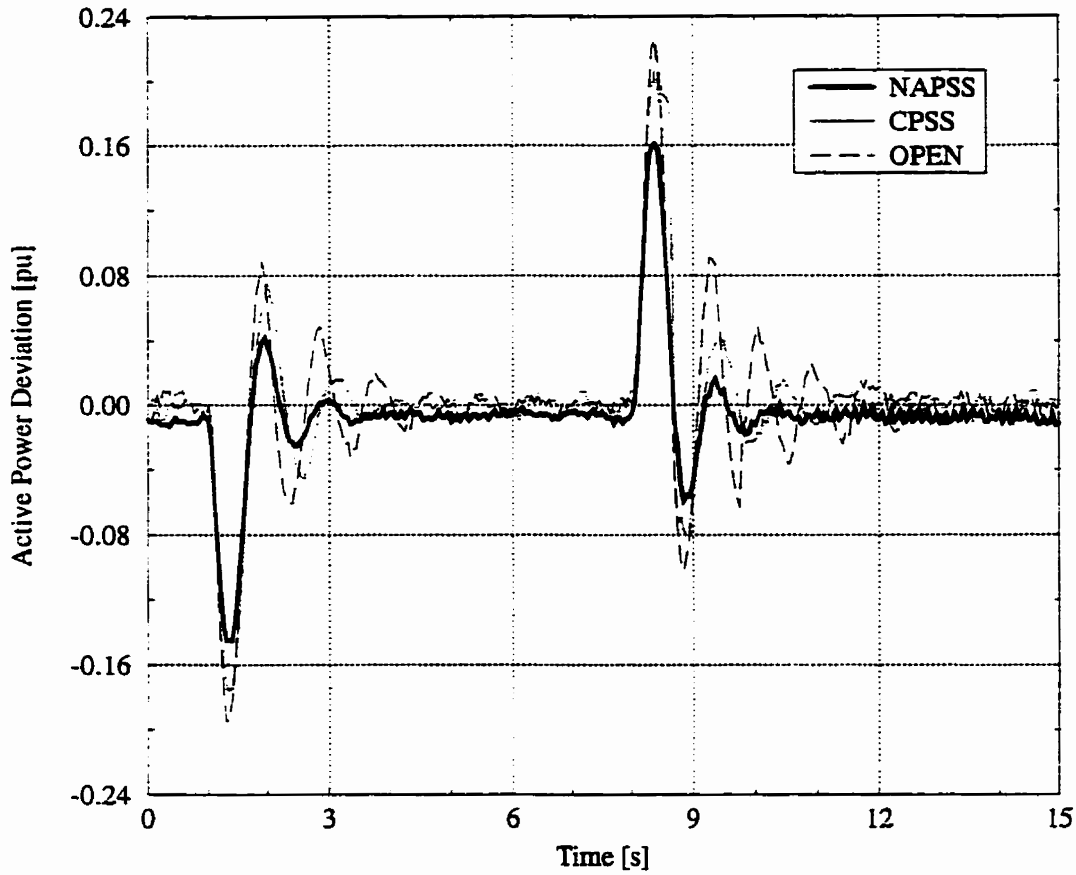


Figure 7.8: System response to a 0.25 *pu* step disturbance in input torque reference, P=0.9 *pu* and pf=0.85 *lag*.

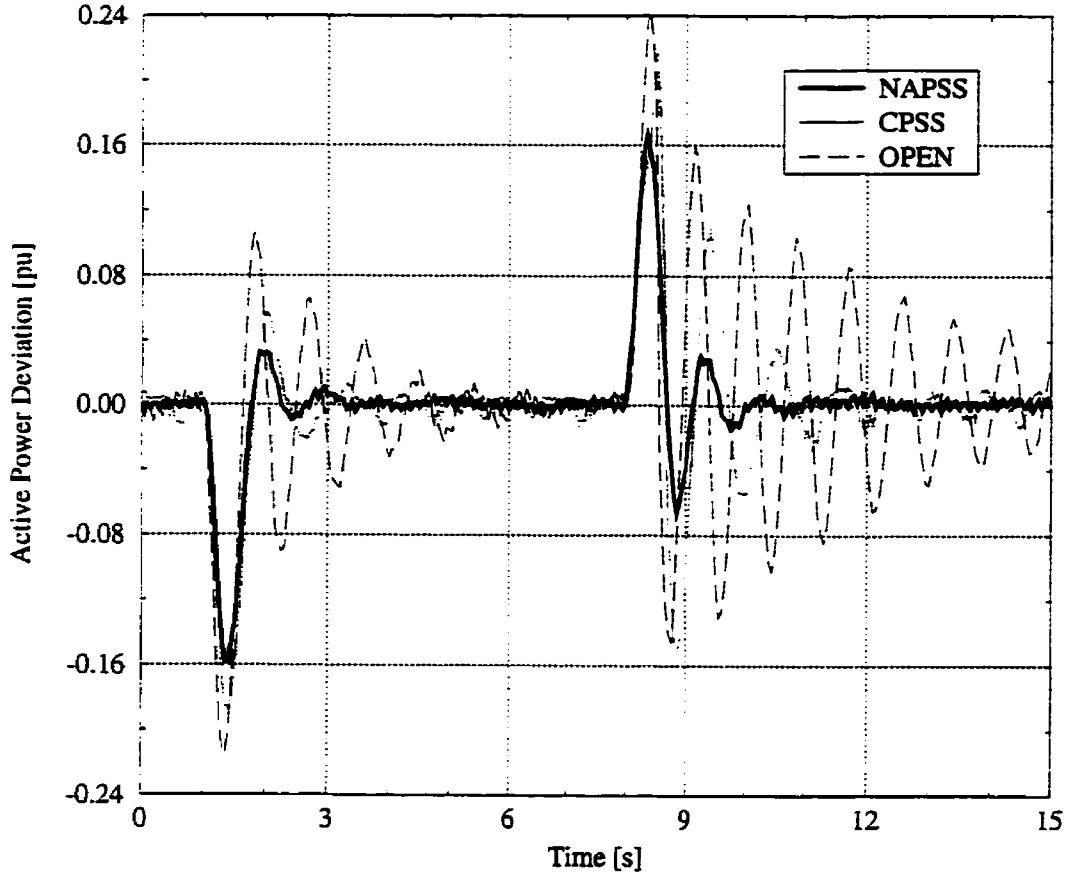


Figure 7.9: System response to a 0.22 *pu* step disturbance in input torque reference, P=0.82 *pu* and pf=0.96 *lead*.

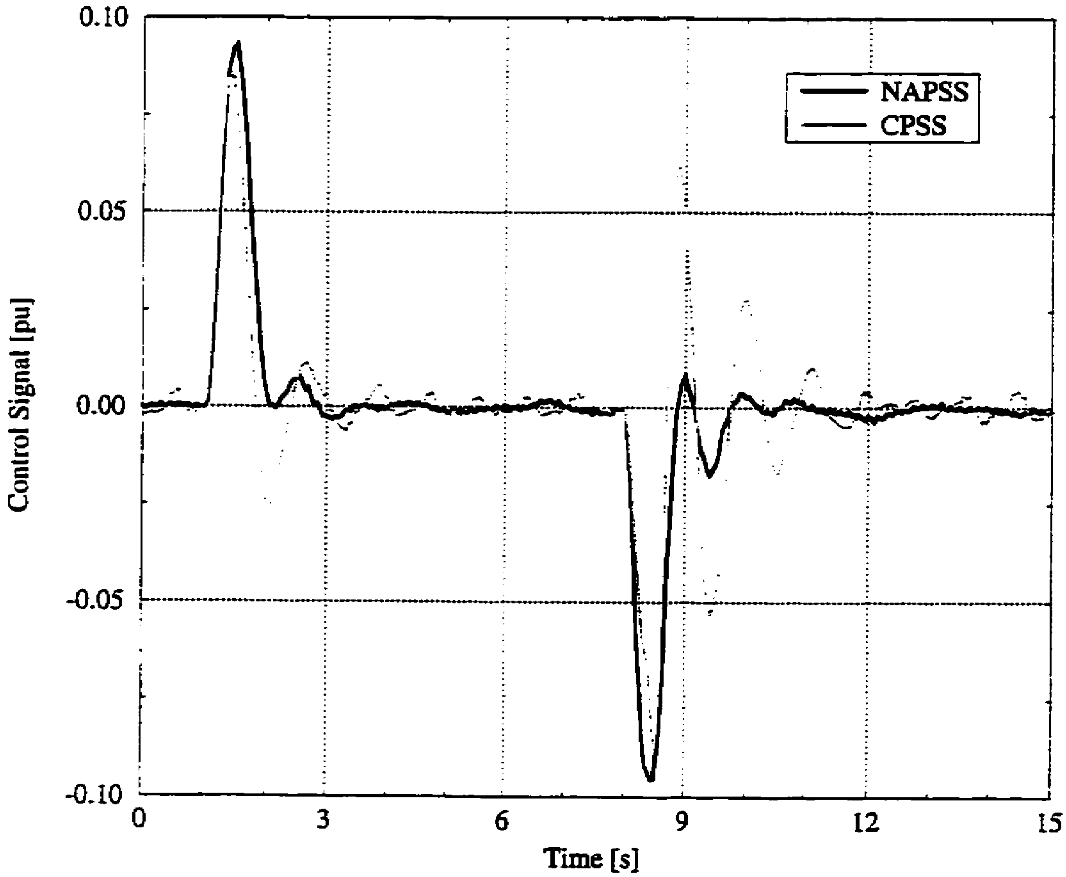


Figure 7.10: Control signals of NAPSS and CPSS to a 0.22 *pu* step disturbance in input torque reference, P=0.82 *pu* and pf=0.96 *lead*.

ditions, a three-phase to ground fault test is conducted. The generator is operating at 0.9 *pu* power, 0.85 *pf lag* and 1.1 *pu* terminal voltage. At 1 *s*, a three-phase to ground fault is applied at the middle of one transmission line. The faulted line is isolated 100 *ms* later by relay action. An unsuccessful reclosure attempt is made 600 *ms* later, and the line is opened again 100 *ms* later due to a permanent fault. The second reclosure is successfully applied at 8 *s* by which the system returns to its original condition. System response under above transient conditions is shown in Fig. 7.11. It is observed that in spite of a large disturbance in the system, the NAPSS manages to control the system properly and damp out the oscillations very effectively.

In another test, the same disturbance is applied with the new operating point of 0.9 *pu* power, 0.95 *pf lead* and 1.0 *pu* terminal voltage. The system response is shown in Fig. 7.12. It is clearly seen in Figs. 7.11 and 7.12 that the NAPSS produces much better results in response to short circuit.

7.7.4 Stability Margin Test

Power system stabilizers are primarily used to provide extra damping to generating units to damp out low frequency oscillations, and thus increase the stability margin of the power system. With PSS in operation, a power system can operate in some overload condition even if it is not stable without a PSS or with a poor PSS. The better the PSS, the more the stability margin is improved. The goal of this test is to observe the ability of NAPSS in enhancing stability margin.

The test starts with the generating unit operating at a stable condition without any PSS. The NAPSS is then switched on to the system. The load

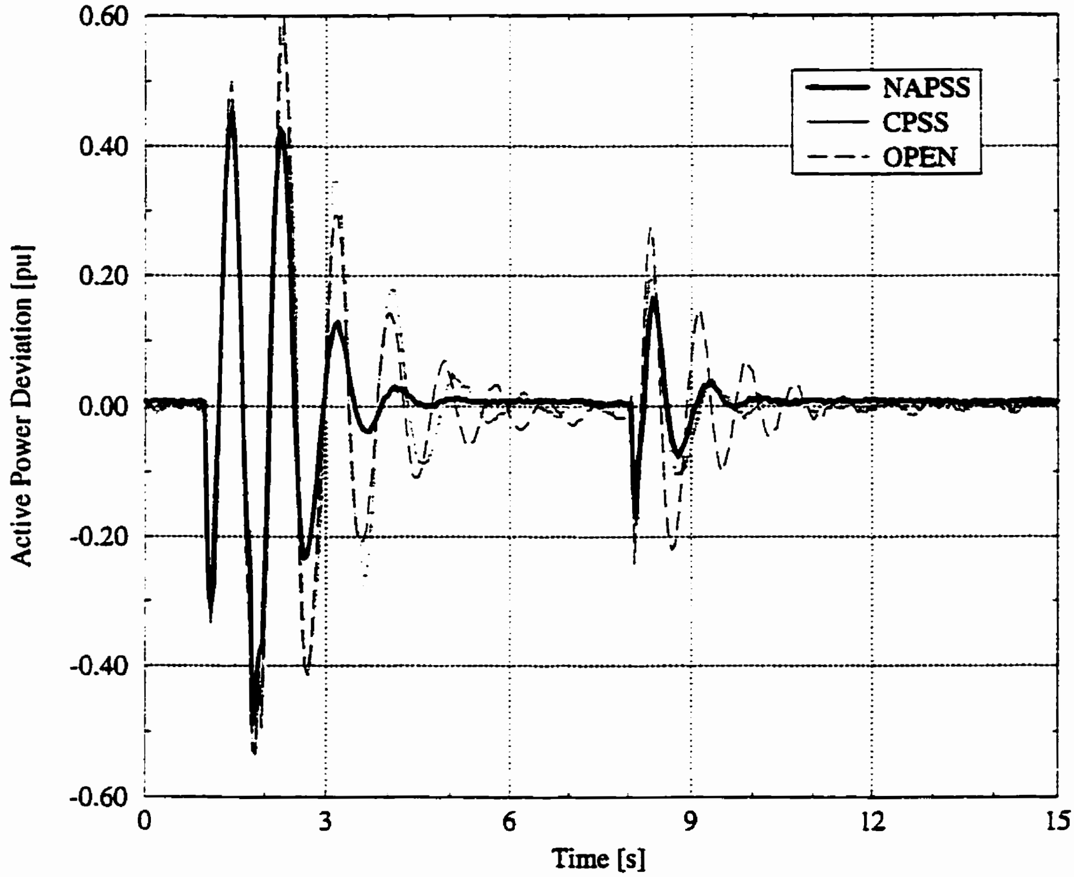


Figure 7.11: System response to a three-phase to ground fault, P=0.9 *pu* and pf=0.85 *lag*.

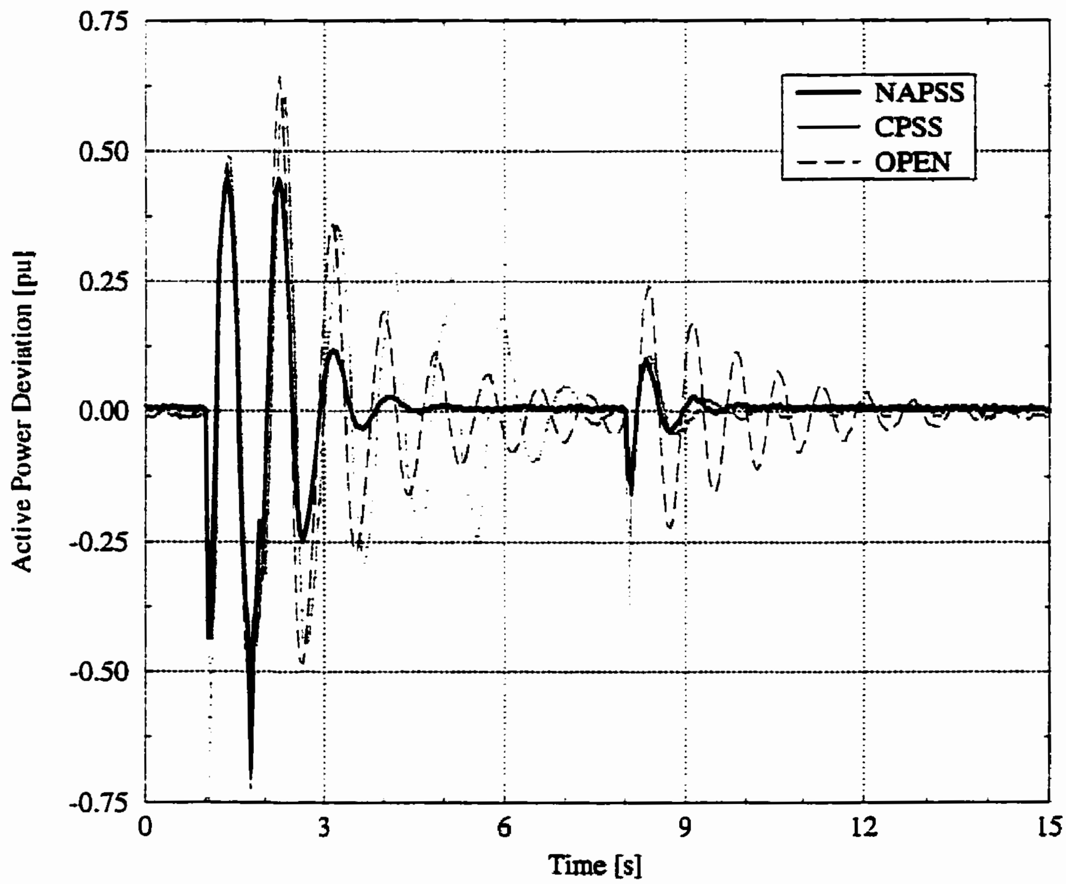


Figure 7.12: System response to a three-phase to ground fault, P=0.9 *pu* and pf=0.95 *lead*.

is increased gradually to a 20% overload condition, i.e. 1.2 *pu* power, 0.9 *pf lag* and 1.05 *pu* terminal voltage. Under this operating condition, the system is still stable as shown in Fig. 7.13. At 5 *s*, the NAPSS is replaced by CPSS. The system begins to oscillate without any external disturbance. This means that the CPSS is unable to maintain system stability for this operating conditions. At 18 *s*, the NAPSS is switched back by which the system is very quickly stabilized. The test proves that the NAPSS can provide a larger stability margin than that of the CPSS.

7.8 Summary

Implementation of NAPSS in a laboratory environment and real-time test results on a physical model power system are presented in this chapter. An experimental physical power system was set up to model a simple power system. This system consists of a micro-alternator driven by a DC motor and a double circuit transmission line linking the micro-alternator to a constant voltage bus. The control strategy and digital control system setup are also discussed.

A sequential implementation method is used to simulate the parallel mechanism of a real multi-layer neural network. The NAPSS is implemented in a real-time digital control environment which was developed using a DSP board and a PLC acting as an AVR. Using system parameters, the NAPSS is first trained using computer simulation. After this stage, the weights of the neural networks are plugged into the real system for further training.

The proposed NAPSS enjoys the adaptation property which is of great importance when operating conditions are changed. The experimental results

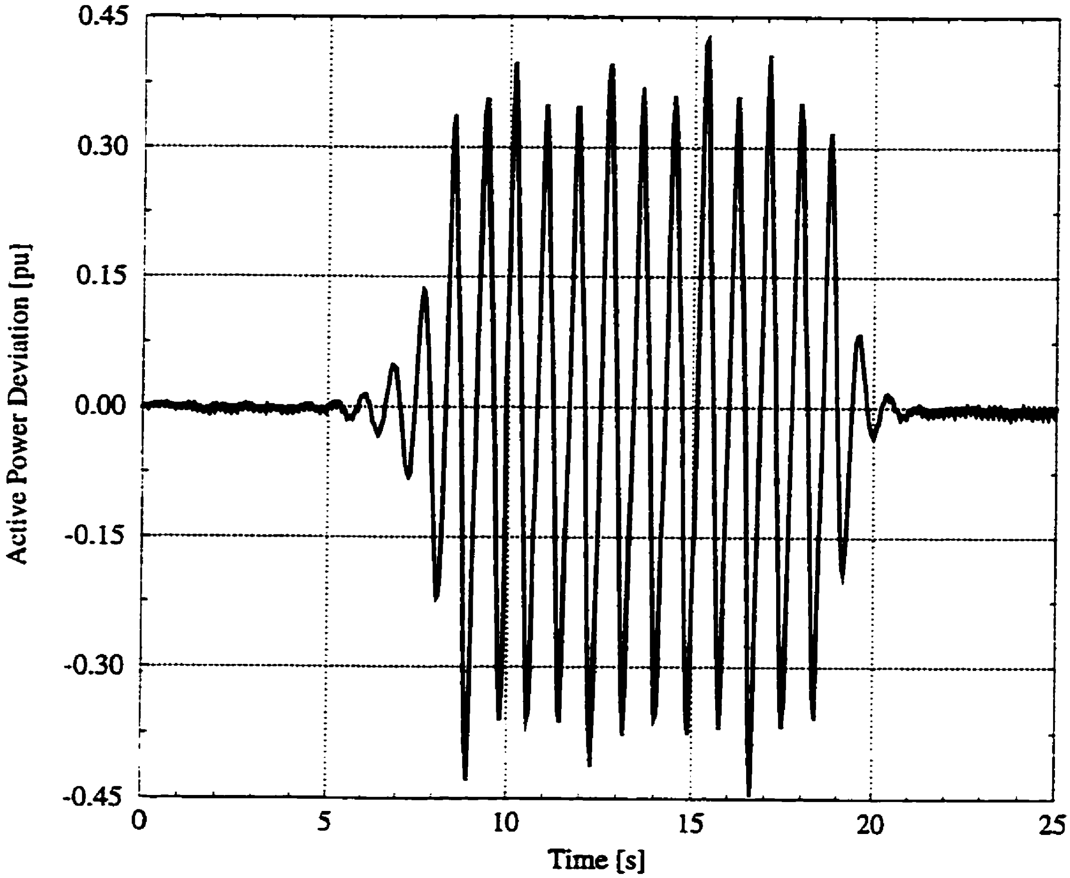


Figure 7.13: Stability margin test.

are compared to those of a digital CPSS. It is demonstrated that NAPSS outperforms the CPSS specially for those operating points which are far from the CPSS design point as well as for large disturbances. It is also shown that using NAPSS, the stability margin of the system is increased.

Chapter 8

Conclusions

8.1 Summary

As discussed in Chapter 1, power system stabilizers have been proven very effective in enhancing stability of power systems. Numerous theoretical studies and experimental tests have been conducted to better understand the behavior of the PSS and to make them more applicable in practice. Different types of PSSs have been investigated, and their advantages and disadvantages have become more and more clear. Based on these studies criteria have been developed to help the designer to choose the most suitable configuration for a particular application.

The conventional PSS has been successfully applied to the power industry in many cases. However, because of its inherent characteristics, it faces some serious problems. Fixed parameter and linear properties are the two most serious problems, since power systems are non-linear time-varying stochastic systems. The stabilizer should be able to adapt itself to the varying system to produce better performance. This has led to research on adaptive power system stabilizers.

This dissertation is devoted to the development of an adaptive power system stabilizer based on on-line trained neural networks. It has made systematic contributions to all three stages of developing such a stabilizer namely theoretical development, simulation studies and experimental tests.

This dissertation begins with the classification and analysis of different types of neural networks. The suitability of different types of NNs for application in power system stability control is investigated. The conclusion that the multi-layer neural network is the one that is most appropriate for application in power system stability control has been reached. Details of the theory and structure of some of the well-known architectures of NN are discussed with the emphasis on the multi-layer neural network and back-propagation learning algorithm. The advantages and disadvantages of using neural networks are presented.

In the first stage, the NAPSS is designed using multi-layer neural networks. The back-propagation algorithm in on-line mode is used to give the powerful property of adaptation to the PSS. The training algorithm is simplified using a scalar error vector. Attempts have been made to make the structure of the NAPSS as simple as possible. Therefore, based on trial and error, two simple networks each having a total of 9 neurons have been chosen, one acting as the identifier and the other acting as the controller. In order to further simplify the implementation of NAPSS in a real situation, the design procedure only uses the output of the generating unit and does not require measurement of the internal states of the plant. Moreover, there is no need for a reference model or teacher signal in the system. The NAPSS is adapted based on the output performance of the system. In other words, it is self-tuning and not model-based. Last but not the least, the NAPSS considers the non-linear nature of the plant as opposed to the CPSS which is based on the linearized model of the plant.

Based on the proposed method, the NAPSS is designed and tested in the single-machine infinite-bus environment by computer simulation [80], [81]. The architecture and parameters of the NAPSS are discussed. Steps of the on-line training algorithm are also explained. Simulation results show that the proposed NAPSS can provide good damping of the power system oscillations over a wide operating range, and significantly and adaptively improve the system performance.

Multi-mode oscillations often occur in a multi-machine power system in which the interconnected generating units have quite different inertias and they are weakly connected by transmission lines. The effectiveness of the NAPSS to damp out multi-mode oscillations in a multi-machine environment is verified in this dissertation. Test results show that each NAPSS can damp out the specific mode of oscillation introduced mainly by the generating unit on which it is applied. Several NAPSSs working together can damp out both the local and the inter-area modes of oscillations. The tests also show that NAPSS can work cooperatively with other types of PSSs [103].

Neural network applications have spread to many fields. Many theoretical investigations have been conducted on NN applications in control systems, but real-time applications are rare. The proposed on-line trained NAPSS is first implemented in this dissertation. Using a micro-alternator, a PLC as AVR, and a DSP board, a real-time digital control environment has been established to test the performance of the NAPSS on a physical system in real-time. A digital CPSS has also been implemented in order to make a comparison between these two approaches. Experimental tests have produced results consistent with simulation studies, proving that the NAPSS has a very good control performance in damping power system low frequency oscillations [106].

8.2 Future Work

Research on neural networks has advanced very fast in recent years, and so have neural network based control techniques. Based on the work of this dissertation, the following are recommended as further research topics:

- Integrated adaptive control of the generating unit for both the excitation and the governor is an area worth looking into. How to consider the interaction between these two signals, and how to use them to produce better results is an interesting topic.
- Integration of the AVR and PSS control loops using neural networks is another topic which seems very promising. The combined controller can get the terminal voltage along with the power deviation as input and produce the field signal as output.
- Laboratory implementation of the NAPSS was based on a sequential simulation method. Since this method uses a sequential method to simulate the parallel distributed nature of the neural networks, computing time of this simulated NN was longer than that of a real NN chip. It

is suggested that the NAPSS be implemented on a commercially available NN chip. It will reduce the sampling time, and thus improve the performance of the NAPSS.

- As for the practical application, there are still many aspects that need to be investigated before the NAPSS can be put into use. For instance, reliability and ability to handle emergencies are the most important aspects to be considered.

Bibliography

[1] J. M. Undrill and J. L. Woodward, "Nonlinear hydro-governing model and improved calculation for determining temporary droop", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-86, no. 4, pp. 443–453, Apr. 1967.

[2] F. P. deMello, "Power system dynamics - overview", *IEEE Power Engineering Society*, Winter Meeting 1975.

[3] C. Concordia, "Appropriate component representation for the simulation of power system dynamics", *IEEE Power Engineering Society*, Winter Meeting 1975.

[4] P. M. Anderson and A. A. Fouad, *Power System Control and Stability*, Iowa State University Press, Ames, Iowa, 1977.

[5] 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, pp. 2211–2218, Sep. 1971.

[6] A. Chandra, *Self-tuning AVR-GUM-stabilizer*, PhD thesis, University of Calgary, 1987.

[7] 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, pp. 316–329, 1969.

[8] F. P. deMello and C. A. Concordia, "Concepts of synchronous machine stability as affected by excitation control", *IEEE Trans. Power Apparatus and Systems*, vol. PAS-88, no. 4, pp. 316–329, Apr. 1969.

[9] F. P. deMello and T. F. Laskowski, "Concepts of power system dynamic stability", *IEEE Trans. Power Apparatus and Systems*, vol. PAS-94, no. 3, pp. 827–833, May/June 1975.

[10] O. L. Elgerd, *Electric Energy Systems Theory - An Introduction*, McGraw-Hill, 1985.

[11] P. O. Bobo, J. W. Skooglund, and C. L. Wagner, "Performance of excitation system under abnormal conditions", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-87, pp. 547–553, Feb. 1968.

[12] J. L. Dineley, A. J. Morris, and C. Preece, "Optimal transient stability from excitation control of synchronous generators", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-87, pp. 1696–1705, Aug. 1968.

[13] P. C. Krause and J. N. Towel, "Synchronous machine damping by excitation control with direct and quadrature axis field windings", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-88, pp. 1266–1274, Aug. 1969.

[14] R. T. Byerly, F. W. Keay, and J. W. Skooglund, "Damping of power oscillations in salient-pole machines with static exciters", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-89, pp. 1009–1021, Jul./Aug. 1970.

[15] P. Subramaniam and O. P. Malik, "Closed loop optimization of power systems with two-axis excitation control", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-92, pp. 167–176, Jan./Feb. 1973.

[16] O. P. Malik, G. S. Hope, and D. W. Huber, "Design and test results of a software based digital avr", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-95, pp. 634–642, Mar./Apr. 1976.

[17] R. Doraiswami, A. M. Sharaf, and J. C. Castro, "A novel excitation control design for multi-machine power system", *IEEE Trans.*

on Power Apparatus and Systems, vol. PAS-103, pp. 1052–1058, May 1984.

[18] P. G. Brown, F. P. deMello, E. H. Lenfest, and R. J. Mills, "Effects of excitation turbine energy control and transmission on transient stability", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-89, pp. 1247–1252, Jul./Aug. 1970.

[19] 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, pp. 1306–1313, May 1968.

[20] S. N. Iyer and B. J. Cory, "Optimal control of a turbo-generator including an exciter and governor", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-90, pp. 2142–2149, Sep./Oct. 1971.

[21] W. A. Mittelstadt, "Four methods of power system damping", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-87, pp. 1323–1329, May 1968.

[22] O. J. Smith, "Power system transient control by capacitor switching", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-88, pp. 28–35, Jan. 1969.

[23] E. W. Kimbark, "Improvement of power system stability by changes in the network", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-88, pp. 773–781, May 1969.

[24] W. A. Mittelstadt and J. L. Saugen, "A method of improving power system transient stability using controllable parameters", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-89, pp. 23–27, Jan. 1970.

[25] P. K. Dash, B. Puthal, O. P. Malik, and G. S. Hope, "Transient stability and optimal control of parallel ac-dc power systems", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-95, pp. 811–820, May/June 1976.

[26] 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, pp. 1426–1434, June 1968.

[27] J. P. Bayne, P. Kundur, and W. Watson, "Static excitation control to improve transient stability", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-94, pp. 1141–1146, July/Aug. 1975.

[28] L. J. Jacovides and B. Adkins, "Effect of excitation regulation on synchronous machine stability", *Proceedings IEE*, vol. 113, pp. 1021–1034, June 1966.

[29] P. L. Dandeno, A. N. Karas, and K. R. McClymont, "Effect of high speed rectifier excitation system on generator stability limits", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-87, pp. 190–200, Jan. 1968.

[30] M. M. El-Metwally, 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, pp. 1192–1200, July/Aug. 1975.

[31] M. Ramamoorty and M. Arumugam, "Design of optimal regulator for synchronous machines", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-92, pp. 269–277, Jan./Feb. 1973.

[32] Y. N. Yu and K. Vongsuriya, "Application of an optimal control theory to a power system", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-89, pp. 55–62, Jan. 1970.

[33] Y. N. Yu and C. Siggers, "Stabilization and optimal control signals for a power system", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-90, pp. 1469–1481, July/Aug. 1971.

[34] D. H. Kelly and M. A. Rahim, "Closed-loop optimal excitation control for power system stability", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-90, pp. 2135–2141, Sep./Oct. 1971.

[35] 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, pp. 2860–2866, Aug. 1982.

[36] 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, pp. 199–211, Jan./Feb. 1973.

[37] E. V. Larsen and D. A. Swann, "Applying power system stabilizers: Part 1-3", *IEEE Trans. Power Apparatus and Systems*, vol. PAS-100, no. 6, pp. 3017–3046, June 1981.

[38] R. G. Farmer, "State-of-the-art technique for system stabilizer tuning", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-102, pp. 699–709, 1983.

[39] K. Bollinger, A. Laha, Hamilton R, and T. Harris, "Power stabilizer design using root locus method", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-94, pp. 1484–1484, 1975.

[40] M. Mobarak, D. Thorne, and E. Hill, "Optimization of exciter and stabilizer parameters of a large thermal generator station", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-97, pp. 1182–1193, 1978.

[41] A. Doi and S. Abe, "Coordinated synthesis of power system stabilizer in multi-machine power system", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-103, pp. 1473–1479, June 1984.

[42] R. J. Fleming, M. A. Mohan, and K. Parvatisam, "Selection of parameters of stabilizers in multi-machine power systems", *IEEE Trans. Power Apparatus and Systems*, vol. PAS-100, no. 5, pp. 2329–2333, May 1981.

[43] 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, pp. 1738–1746, 1983.

[44] D. A. Pierre, "A perspective on adaptive control of power systems", *IEEE Trans. on Power Systems*, vol. PWRS-2, pp. 387–396, 1987.

[45] T. Hiyama and T. Sameshima, "Fuzzy logic control scheme for on-line stabilization of multi-machine power system", *Fuzzy Sets and Systems*, vol. 39, pp. 181–194, 1991.

[46] M. A. Hassan, O. P. Malik, and G. S. Hope, "A fuzzy logic based stabilizer for a synchronous machine", *IEEE Trans. Energy Conversion*, vol. 6, no. 3, pp. 407–413, 1991.

[47] K. A. El-Metwally and O. P. Malik, "Fuzzy logic power system stabilizer", *IEE Proc. on Generation, Transmission and Distribution*, vol. 142, no. 3, pp. 277–281, May 1995.

[48] A. Ghosh, G. Ledwich, O. P. Malik, and G. S. Hope, "Power system stabilizer based on adaptive control techniques", *IEEE Trans. on Power Apparatus and Systems*, vol. PAS-103, pp. 1983–1986, 1984.

[49] S. J. Cheng, O. P. Malik, and G. S. Hope, "Damping of multi-mode oscillations in power system using a dual-rate adaptive stabilizer", *IEEE Trans. on Power Systems*, vol. PWRS-3, no. 1, pp. 101–108, 1988.

[50] G. P. Chen, O. P. Malik, G. S. Hope, Y. H. Qin, and G. Y. Xu, "An adaptive power system stabilizer based on the self-optimizing pole-shifting control strategy", *IEEE Trans. Energy Conversion*, vol. 8, no. 4, pp. 639–645, 1993.

[51] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity", *Bulletin of Mathematical Biophysics*, vol. 9, pp. 127–147, 1943.

[52] D. O. Hebb, *The Organization of Behavior*, Wiley, New York, 1949.

[53] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain", *Psychological Review*, vol. 65, pp. 386–408, 1958.

[54] M. L. Minsky and S. A. Papert, *Perceptrons*, MIT Press, Cambridge, MA, 1969.

[55] S. Grossberg, "Adaptive pattern classification and universal recording: I. parallel development and coding of neural feature detectors", *Biological Cybernetics*, vol. 23, pp. 121–134, 1976.

[56] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities", *Proceedings of the National Academy of Science*, vol. 79, pp. 2554–2558, 1982.

[57] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. 1: Foundations, MIT Press, Cambridge, MA, 1986.

[58] R. P. Lippman, "An introduction to computing with neural nets", *IEEE Acoustics, Speech and Signal Processing Magazine*, vol. 2, pp. 4–22, Apr. 1987.

[59] D. J. Sobajic and Y. H. Pao, "Artificial neural-net based dynamic security assessment for electric power systems", *IEEE Trans. on Power Systems*, vol. PWRS-4, pp. 220–228, Feb. 1989.

[60] V. S. S. Vankayala and N. D. Rao, "Artificial neural networks and their application to power system - a bibliographical survey", *Electric Power Systems Research*, vol. 28, no. 1, pp. 67–79, Oct. 1993.

[61] P. J. Antsaklis, "Neural networks in control systems", *IEEE Control Systems Magazine*, vol. 12, pp. 8–10, Apr. 1992.

[62] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Macmillan, New York, 1994.

[63] N. Wiener, *Cybernetics: or control and communications in the animal and the machine*, MIT Press, Cambridge, MA, 1948.

[64] B. Widrow, "The original adaptive broom-balancer", in *Proceedings of IEEE Int. Symposium on Circuits and Systems*, 1987, pp. 351–357.

[65] H. Tolle and E. Ersu, *Lecture Notes in Control and Information Sciences*, chapter Neurocontrol: Learning control systems inspired by neural architectures and human problem solving, Springer Verlag, Berlin, 1992.

[66] W. T. Miller, R. S. Suttin, and P. J. Werbos, *Neural Networks for Control*, MIT Press, Cambridge, MA, 1990.

[67] S. Chen and S. A. Billings, *Advances in Intelligent Control*, chapter 4: Neural networks for non-linear dynamic system modeling and identification, Taylor and Francis, London, 1994.

[68] Y. H. Pao, S. M. Phillips, and D. J. Sobajic, *Advances in Intellingent Control*, chapter 3: Neural-net computing and intelligent control of systems, Taylor and Francis, London, 1994.

[69] J. Moody, *Advances in Neural Information Processing Systems*, chapter Fast learning in multi-resolution hierarchies, pp. 29–39, Moragn Kaufmann, San Mateo, CA, 1989.

[70] B. Widrow and M. A. Lehr, "30 years of adaptive neural networks: Perceptron, madaline and back-propagation", *Proceedings of IEEE*, vol. 78, pp. 1415–1442, Sep. 1990.

[71] B. Widrow and M. E. Hoff, "Adaptive switching circuits", *IRE WESCON Convention Record, New York: IRE*, pp. 96–104, 1960.

[72] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. 1: Foundations, chapter 8, pp. 318–362, MIT Press, Cambridge, MA, 1986.

[73] P. J. Werbos, *Beyond regression: New tools for prediction and analysis in behavioral sciences*, PhD thesis, Harvard University, 1974.

[74] N. Kamiyama, H. Watanabe, N. Iijima, H. Mitsui, Y. Yoshida, and M. Sone, "Tuning of learning rate and momentum on back-propagation", in *Proceedings of Second IASTED International Conference: Computer Applications in Industry*, Alexandria, Egypt, May 1992, pp. 98–101.

[75] J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons", *Proceedings of the National Academy of Science*, vol. 81, pp. 3088–3092, 1984.

[76] T. Kohonen, *Self-Organization and Associative Memory*, Springer Verlag, Berlin, 1989.

[77] K. S. Narendra and A. M. Annaswamy, *Stable Adaptive Systems*, Prentice Hall, Englewood Cliffs, NJ, 1989.

[78] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks", *IEEE Trans. Neural Networks*, vol. 1, pp. 4–27, Mar. 1990.

[79] P. Shamsollahi and O. P. Malik, "Direct neural adaptive control applied to synchronous generator", in *1997 IEEE International Electric Machines and Drives Conference Record, IEEE-IEMDC'97*, Milwaukee, May 1997, paper no. MB3-2.

[80] P. Shamsollahi and O. P. Malik, "An adaptive power system stabilizer using on-line trained neural networks", *presented at 1997 IEEE Power Engineering Society Winter Meeting*, New York, Feb. 1997, To be published in *IEEE Trans. on Energy Conversion*.

[81] P. Shamsollahi, J. He, and O. P. Malik, "A self-learning controller to enhance power system stability using feed-forward and recurrent neural networks", *Canadian Journal of Electrical and Computer Engineering*, Submitted in Sep. 1996.

[82] D. R. Hush and B. G. Horne, "Progress in supervised neural networks - what's new since lippmann", *IEEE Signal Processing Magazine*, pp. 8–39, Jan. 1993.

[83] G. Cybenko, "Approximation by superposition of a sigmoidal function", *Math. Control, Signals, Systems*, vol. 2, pp. 303–314, 1989.

[84] P. J. Werbos, "Back propagation through time: What it does and how to do it", *Proc. IEEE*, vol. 78, pp. 1550–1560, Oct. 1990.

[85] S. Chen and S. A. Billings, "Nonlinear system identification using neural networks", *Int. J. of Control*, vol. 51, pp. 1191–1214, 1990.

[86] K. S. Narendra and K. Parthasarathy, "Neural networks and dynamical systems", *Int. J. of Approximate Reasoning*, vol. 6, no. 2, pp. 109–131, 1992.

[87] P. Shamsollahi and O. P. Malik, "On-line identification of synchronous generator using neural networks", in *Proceedings of the Canadian Conference on Electrical and Computer Engineering*, Calgary, May 1996, pp. 595–598.

[88] B. Srinivasan and U. R. Prasad, "Back propagation through adjoints for the identification of nonlinear dynamic systems using recurrent neural models", *IEEE Trans. Neural Networks*, vol. 5, no. 2, pp. 213–227, Mar. 1994.

[89] D. Nguyen and B. Widrow, "The truck backer-upper: An example of self-learning in neural networks", in *Proceedings of the Int. Joint Conference on Neural Networks*, Washington DC, June 1989, vol. 2, pp. II357–II363.

[90] Yao-Nan Yu, *Electric Power System Dynamics*, Academic Press, New York, 1983.

[91] D. C. Lee, R. E. Beaulieu, and J. R. R. Service, "A power system stabilizer using speed and electrical power input - design and field experience", *IEEE Trans. Power Apparatus and Systems*, vol. PAS-100, no. 9, pp. 4151–4157, Sept. 1981.

[92] M. Mohamed, D. Thorn, and E. Hill, "Contrast of power system stabilizer performance on hydro and thermal units", *IEEE Trans. Power Apparatus and Systems*, vol. PAS-99, pp. 1522–1533, 1980.

[93] Y. Zhang, G. P. Chen, O. P. Malik, and G. S. Hope, "An artificial neural network based adaptive power system stabilizer", *IEEE Trans. Energy Conversion*, vol. 8, no. 1, pp. 71–77, Mar. 1993.

[94] Q. H. Wu, B. W. Hogg, and G. W. Irwin, "A neural network regulator for turbo-generators", *IEEE Trans. Neural Networks*, vol. 3, no. 1, pp. 95–100, Jan. 1992.

[95] L. Guan, S. Cheng, and R. Zhou, "Artificial neural network power system stabilizer trained with an improved bp algorithm", *IEE Proceedings-C*, vol. 143, no. 2, pp. 135–141, Mar. 1996.

[96] Y. M. Park and K. Y. Lee, "A neural network-based power system stabilizer using load flow characteristics", *IEEE Tran. Energy Conversion*, vol. 11, no. 2, pp. 435–441, June 1996.

[97] Y. Zhang, *Artificial neural network based PSS*, PhD thesis, University of Calgary, 1994.

[98] T. Kobayashi and A. Yokoyama, "An adaptive neuro-control system of synchronous generator for power system stabilization", *IEEE Trans. on Energy Conversion*, vol. 11, no. 3, pp. 621–630, Sept. 1996.

[99] K. S. Narendra and K. Parthasarathy, "Gradient methods for the optimization of dynamical systems containing neural networks", *IEEE Trans. Neural Networks*, vol. 2, no. 2, pp. 252–262, Mar. 1991.

[100] P. Shamsollahi and O. P. Malik, "Design of a neural adaptive power system stabilizer using dynamic back-propagation method", *Electric Power System Research Journal*, Submitted in Jan. 1997.

[101] P. Shamsollahi, "Application of indirect neural adaptive control to multi-machine power systems using dynamic bp method", in *Proceeding of the Student Paper/Poster Session of the IEEE-PES 20th International Conference on Power Industry Applications, PICA'97*, Columbus, May 1997, pp. 97–101.

[102] IEEE Standard 421.5, *IEEE Recommended Practice for Excitation Systems for Power System Stability Studies*, 1992.

[103] P. Shamsollahi and O. P. Malik, "Application of neural adaptive power system stabilizer in a multi-machine power system", *IEEE Power Engineering Society, Winter Meeting 1998*, Submitted in Jan. 1997.

[104] S. Weerasooriya and M. A. El-Sharkawi, "Laboratory implementation of a neural network trajectory controller for a dc motor", *IEEE Trans. on Energy Conversion*, vol. 8, pp. 107–113, Mar. 1993.

[105] Y. Zhang and O. P. Malik, "Experimental studies with a neural network based power system stabilizer", in *Proceedings of IEEE Int. Conference on Intelligent Systems Applications to Power Systems*, Orlando, FL, 1996, vol. 1, pp. 104–108.

[106] P. Shamsollahi and O. P. Malik, "Real-time implementation and experimental studies of a neural adaptive power system stabilizer", *IEEE Power Engineering Society, Winter Meeting 1998*, Submitted in May. 1997.

Appendix A

Single-Machine Power System

A.1 Synchronous Generator Model

The generating unit is modeled by seven first-order differential equations given below

$$\dot{\delta} = \omega_0 \omega \tag{A.1}$$

$$\dot{\omega} = \frac{1}{2H}(T_m + g + K_d \dot{\delta} - T_e) \tag{A.2}$$

$$\dot{\lambda}_d = e_d + r_a i_d + \omega_0(\omega + 1)\lambda_q \tag{A.3}$$

$$\dot{\lambda}_q = e_q + r_a i_q - \omega_0(\omega + 1)\lambda_d \tag{A.4}$$

$$\dot{\lambda}_f = e_f - r_f i_f \tag{A.5}$$

$$\dot{\lambda}_{kd} = -r_{kd} i_{kd} \tag{A.6}$$

$$\dot{\lambda}_{kq} = -r_{kq} i_{kq} \tag{A.7}$$

A.2 Governor

The governor employed in the simulation study has the transfer function

$$g(s) = \left[a + \frac{b}{1 + sT_g}\right] \dot{\delta}(s)$$

A.3 Power System Parameters

Power system parameters used in the simulation study are given below

$H = 3.46$	$r_a = 0.007$	$r_f = 0.00089$
$r_{kd} = 0.023$	$r_{kq} = 0.023$	$r_t = 0.05$
$x_d = 1.24$	$x_{kd} = 1.15$	$x_{md} = 1.126$
$x_q = 0.743$	$x_{kq} = 0.652$	$x_{mq} = 0.626$
$x_f = 1.33$	$x_t = 0.6$	$K_d = -\ 0.027$
$R_C = 0.0$	$X_C = 0.0$	$K_C = 0.08$
$T_B = 10.0$	$T_{B1} = 0.0$	$K_{LR} = 0.0$
$T_C = 1.0$	$T_{C1} = 0.0$	$I_{LR} = 0.0$
$T_A = 0.0$	$K_A = 190$	$T_R = 0.04$
$T_F = 1.0$	$K_F = 0.0$	$V_{OEL} = 999$
$V_{AMIN} = -999$	$V_{AMAX} = 999$	$V_{UEL} = -999$
$V_{IMIN} = -999$	$V_{IMAX} = 999$	$V_{STMIN} = -0.1$
$V_{RMIN} = -6.7$	$V_{RMAX} = 7.8$	$V_{STMAX} = 0.1$
$a = -0.001328$	$b = -\ 0.17$	$T_g = 0.25$
$T_1 = 0.2$	$T_2 = 0.045$	$T_3 = 0.2$
$T_4 = 0.045$	$T_5 = 2.65$	$T_6 = 0.009$
$K_s = 11.31$		

All of the resistances and reactances are in *pu* and the time constants are in seconds.

A.4 Conventional PSS

The CPSS used in the simulation study is Type PSS1A from *IEEE* Standard 421.5 [102] with the following transfer function (no discontinuous excitation control is used)

$$U_{PSS}(s) = V_{ST}(s) = K_s \frac{sT_5}{1 + sT_5} \frac{1 + sT_1}{1 + sT_2} \frac{1 + sT_3}{1 + sT_4} \frac{1}{1 + sT_6} \omega(s) \qquad \text{(A.8)}$$

A.5 AVR and Exciter

The AVR and exciter combination used in the system is from *IEEE* Standard 421.5, Type ST1A [102] as shown in Fig. A.1.

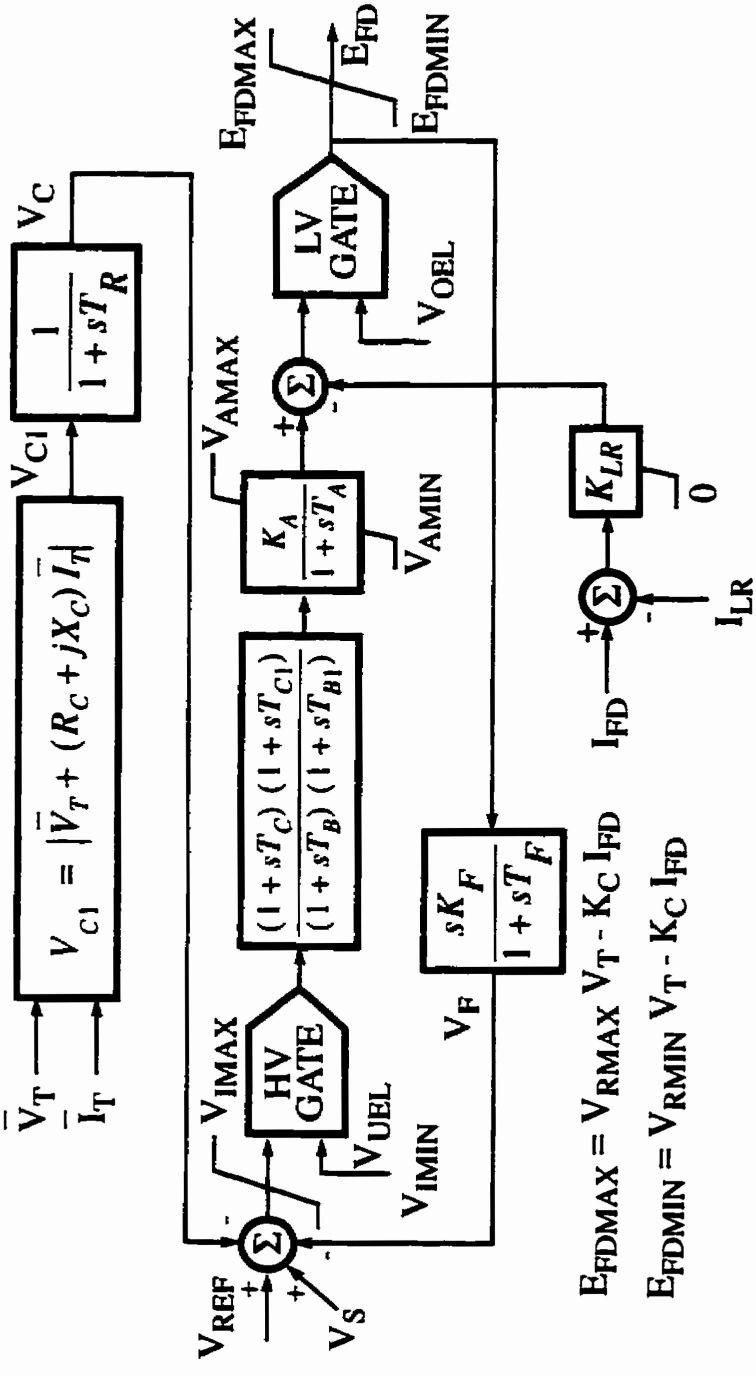


Figure A.1: AVR and exciter model.

Appendix B

MATLAB Function for Controller Training

In order to train the controller, the error should be first back-propagated through the identifier to reach the controller. In this situation, one can assume cascade of the controller and identifier as one neural network in which the error is first back-propagated through the first block (identifier), assuming fixed weights, and then the back-propagated error is further back-propagated through the second block (controller) updating its weights.

The following MATLAB function serves this purpose. The definition of the variables used is coming first.

```
f1_c = controller first  layer neuron function
f2_c = controller second layer neuron function
f1_i = identifier first  layer neuron function
f2_i = identifier second layer neuron function
```

ercn = controller error

a1_i = output of identifier first layer

a2_i = output of identifier second layer (plant's predicted
output at next step)

d1_i = delta-term of the first layer of identifier

d2_i = delta-term of the second layer of identifier

w1_i = identifier first layer weights

w2_i = identifier second layer weights

d1_u = delta-term associated with the u(k) input line of identifier

n_del_omega = total number of omega (speed) input lines for
identifier (=3)

ercn_eq = controller equivalent error (back-propagated through
identifier)

h = tuning parameter

a1_c = output of controller first layer

a2_c = output of controller second layer (control signal)

d1_c = delta-term of the first layer of controller

d2_c = delta-term of the second layer of controller

w1_c = controller first layer weight matrix

w2_c = controller second layer weight matrix

b1_c = controller first layer bias vector

b2_c = controller second layer bias vector

dw1_c = correction to controller first layer weight matrix

dw2_c = correction to controller second layer weight matrix

db1_c = correction to controller first layer bias vector

db2_c = correction to controller second layer bias vector

inp_c = controller input vector

lr_c = controller learning rate

mc_c = controller momentum constant (=0)

```
%-----------------------------------
%     Controller Training Function    |
%-----------------------------------

f1_c = 'tansig';
f2_c = 'purelin';
f1_i = 'tansig';
f2_i = 'purelin';

df1_c = feval(f1_c,'delta');
df2_c = feval(f2_c,'delta');
df1_i = feval(f1_i,'delta');
df2_i = feval(f2_i,'delta');

ercn = -a2_i;
d2_i = feval(df2_i,a2_i,ercn);
d1_i = feval(df1_i,a1_i,d2_i,w2_i);
d1_u = d1_i'*w1_i(:,(n_del_omega+1));

ercn_eq = d1_u + h.*a2_c;
```

```
d2_c     = feval(df2_c,a2_c,ercn_eq);
d1_c     = feval(df1_c,a1_c,d2_c,w2_c);

[dw1_c,db1_c] = learnbpm(inp_c,d1_c,lr_c,mc_c,dw1_c,db1_c);
[dw2_c,db2_c] = learnbpm(a1_c,d2_c,lr_c,mc_c,dw2_c,db2_c);

w1_c = w1_c + dw1_c;
w2_c = w2_c + dw2_c;
b1_c = b1_c + db1_c;
b2_c = b2_c + db2_c;
```

Appendix C

Multi-Machine Power System

C.1 Synchronous Generator Model

The generating unit is modeled by five first order differential equations given below

$$\dot{\delta} = \omega_0 \omega \tag{C.1}$$
$$\dot{\omega} = \frac{1}{2H}(T_m + g + K_d\dot{\delta} - T_e) \tag{C.2}$$
$$T'_{do}\dot{e}'_q = e_f - (x_d - x'_d)i_d - e'_q \tag{C.3}$$
$$T''_{do}\dot{e}''_q = [e'_q - (x'_d - x''_d)i_d - e''_q] + T''_{do}\dot{e}'_q \tag{C.4}$$
$$T''_{qo}\dot{e}''_d = (x_q - x''_q)i_q - e''_d \tag{C.5}$$

C.2 Governors Parameters

	G_1	G_2	G_3	G_4	G_5
T_g	0.2500	0.2500	0.2500	0.2500	0.2500
a	-1.5e-4	-1.5e-4	-1.33e-3	-1.5e-4	-1.33e-3
b	-0.0150	-0.0150	-0.1700	-0.0150	-0.1700

C.3 Parameters of the Generators

	G_1	G_2	G_3	G_4	G_5
x_d	0.1026	0.1026	1.0260	0.1026	1.0260
x_q	0.0658	0.0658	0.6580	0.0658	0.6580
x'_d	0.0339	0.0339	0.3390	0.0339	0.3390
x''_d	0.0269	0.0269	0.2690	0.0269	0.2690
x''_q	0.0335	0.0335	0.3350	0.0335	0.3350
T'_{do}	5.6700	5.6700	5.6700	5.6700	5.6700
T''_{do}	0.6140	0.6140	0.6140	0.6140	0.6140
T''_{qo}	0.7230	0.7230	0.7230	0.7230	0.7230
H	80.000	80.000	10.000	80.000	10.000

C.4 Parameters of AVRs and Exciters

Parameters of AVRs and simplified ST1A exciters [102] are

	G_1	G_2	G_3	G_4	G_5
K_A	190.00	190.00	190.00	190.00	190.00
K_C	0.0800	0.0800	0.0800	0.0800	0.0800
T_B	10.000	10.000	10.000	10.000	10.000
T_C	1.0000	1.0000	1.0000	1.0000	1.0000
T_R	0.0400	0.0400	0.0400	0.0400	0.0400

The output of all exciters is limited within -6.7 to 7.8 pu.

C.5 Transmission Lines Parameters

Bus No.	r_t	x_t	$B_t/2$
1 − 7	0.00435	0.01067	0.01536
2 − 6	0.00213	0.00468	0.00404
3 − 6	0.01002	0.03122	0.03204
3 − 6	0.01002	0.03122	0.03204
4 − 8	0.00524	0.01184	0.01756
5 − 6	0.00711	0.02331	0.02732
6 − 7	0.04032	0.12785	0.15858
7 − 8	0.01724	0.04153	0.06014

C.6 Operating Point #1

	G_1	G_2	G_3	G_4	G_5
$P_e(pu)$	5.1076	8.5835	0.8055	8.5670	0.8501
$Q(pu)$	6.8019	4.3836	0.4354	4.6686	0.2264
$V_t(pu)$	1.0750	1.0500	1.0250	1.0750	1.0250
$\delta(rad)$	0.0000	0.3168	0.2975	0.1174	0.3051

Loads in admittance in *pu*

$$L_1 = 7.5 - j5.0 \quad L_2 = 8.5 - j5.0 \quad L_3 = 7.0 - j4.5$$

C.7 Operating Point #2

	G_1	G_2	G_3	G_4	G_5
$P_e(pu)$	3.1558	3.8835	0.4055	4.0670	0.4501
$Q(pu)$	2.9260	1.4639	0.4331	2.1905	0.2575
$V_t(pu)$	1.0500	1.0300	1.0250	1.0500	1.0250
$\delta(rad)$	0.0000	0.1051	0.0943	0.0361	0.0988

Loads in admittance in *pu*

$$L_1 = 3.76 - j2.5 \quad L_2 = 4.25 - j2.5 \quad L_3 = 3.5 - j2.25$$

Appendix D

Physical Model Power System

D.1 Micro-Alternator Parameters

$$H = 4.75 \quad r_a = 0.0026 \quad r_f = 0.00075 \quad x_f = 1.27$$

$$r_{kd} = 0.0083 \quad x_{kd} = 1.25 \quad x_{md} = 1.129 \quad x_d = 1.2$$

$$r_{kq} = 0.0083 \quad x_{kq} = 1.25 \quad x_{mq} = 1.129 \quad x_q = 1.2$$

D.2 Transmission Line Parameters

Each circuit of transmission line consists of six 50 km equivalent π-sections. Parameters of each section are

$$r_t = 0.036 \quad x_t = 0.0706 \quad B_t = 18.779$$

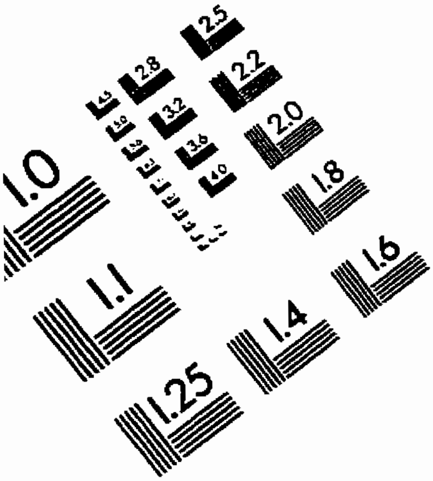
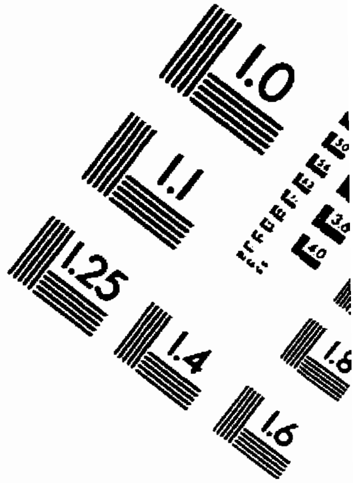
D.3 Conventional PSS Parameters

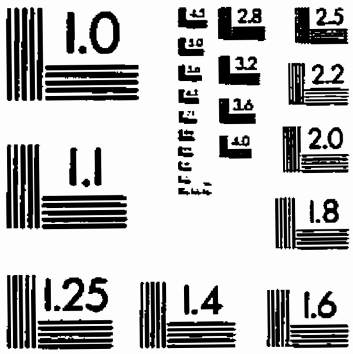
$$K_s = \text{-}9.6 \quad T_1 = 0.1 \quad T_2 = 0.08$$

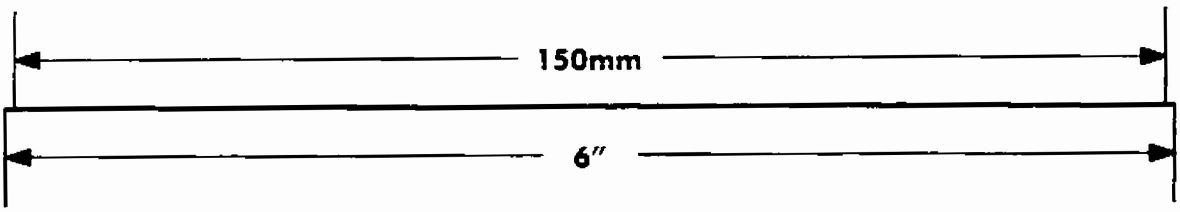
$$T_3 = 0.1 \quad T_4 = 0.08$$

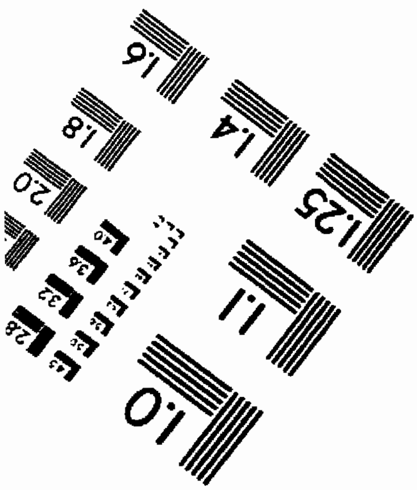
All of the resistances and reactances are in pu and the time constants are in seconds.

IMAGE EVALUATION
TEST TARGET (QA-3)







APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989