Robust methods of control for power system damping
by Fereshteh Fatehi

A thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Electrical Engineering
Montana State University

Abstract:
This thesis is concerned with development of robust controller strategies which can be applied to static var compensators (SVC) in order to dampen low-frequency interarea oscillations in noisy multimachine power systems.

Two robust control strategies, adaptive and nonadaptive, are developed. The adaptive controller includes an on-line real-time recursive extended least-squares identifier to determine parameters of the system and noise model; the model parameters are used in a real-time control-design algorithm to obtain digital controller parameters; and the resulting digital control algorithm uses both system outputs and past control actions to generate control signals for the process. The nonadaptive design technique is based on combined system identification and controller design process. An iterative closed-loop identification method is used to find a linear model for the power system. To make the controller more robust to plant parameter variation and unmodeled dynamics, a linear quadratic Gaussian controller design method with loop transfer recovery (LQG/LTR) based on a generalized technique for the nonminimum phase (NMP) power system model is used to design the controller.

Detailed simulation results are presented that show the feasibility and the properties of both robust control schemes.

ROBUST METHODS OF CONTROL FOR POWER SYSTEM DAMPING

by

Fereshteh Fatehi

A thesis submitted in partial fulfillment
of the requirements for the degree

of

Doctor of Philosophy

in

Electrical Engineering

MONTANA STATE UNIVERSITY
Bozeman, Montana

February 1995

APPROVAL

of a thesis submitted by

Fereshteh Fatehi

This thesis has been read by each member of the thesis committee and has been found to be satisfactory regarding content, English usage, format, citations, bibliographic style, and consistency, and is ready for submission to the College of Graduate Studies.

3/9/95 — Date

Donald A. Pierre
Donald A. Pierre
Chairperson, Graduate Committee

Approved for the Major Department

3/9/95 — Date

Victor Gerez
Head, Electrical Engineering

Approved for the College of Graduate Studies

3/24/95 — Date

R L Brown
Robert Brown
Graduate Dean

STATEMENT OF PERMISSION TO USE

Signature F. Fatehi

Date 3/9/95

ACKNOWLEDGMENTS

I would like to express my gratitude to Professor Donald A. Pierre, my principal advisor, for his guidance, continuous support and constructive criticism throughout my studies and the writing of this thesis. I would also like to thank Dr. James Smith and Dr. Hashem Nehrir for their suggestions and help concerning this research. To the rest of my committee, Dr. Byron Bennett, Dr. Curtis Vogel and Dr. William Hiscock, thank you for the time you spent offering advice when it was needed. I would like to thank Dr. Roy Johnson for the computer support that he has provided. Many others also have been helpful to the work of this thesis including Mr. Stott Woods, Dr. Iraj Sadighi and Dr. Dan Trudnowski whose codes in adaptive control were used in this thesis.

The financial support of the Montana State University Engineering Experiment Station, the Montana Electric Power Research Association, the Bonneville Power Administration and the National Science Foundation is appreciated.

Finally, I would like to express my appreciation to my parents, my sister Fahime, my brothers and my best friend Maryam for their support throughout my studies.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

This thesis is concerned with development of robust controller strategies which can be applied to static var compensators (SVC) in order to dampen low-frequency interarea oscillations in noisy multimachine power systems.

Two robust control strategies, adaptive and nonadaptive, are developed. The adaptive controller includes an on-line real-time recursive extended least-squares identifier to determine parameters of the system and noise model; the model parameters are used in a real-time control-design algorithm to obtain digital controller parameters; and the resulting digital control algorithm uses both system outputs and past control actions to generate control signals for the process. The nonadaptive design technique is based on combined system identification and controller design process. An iterative closed-loop identification method is used to find a linear model for the power system. To make the controller more robust to plant parameter variation and unmodeled dynamics, a linear quadratic Gaussian controller design method with loop transfer recovery (LQG/LTR) based on a generalized technique for the nonminimum phase (NMP) power system model is used to design the controller.

Detailed simulation results are presented that show the feasibility and the properties of both robust control schemes.

CHAPTER 1

INTRODUCTION

Power systems commonly exhibit low-frequency electromechanical oscillations particularly when operated under conditions where large power transfers are being made over long transmission lines. Many types of unavoidable system disturbances can cause electromechanical oscillation, and severe oscillations can decrease the life of generators and limit the amount of transferable power over transmission lines.

There are two types of electromechanical oscillations: local modes (1 to 3 Hz) and interarea modes (0.1 to 1 Hz). A local mode of oscillation occurs when a single generator swings against the system while an interarea mode of oscillation occurs when a group of generators swing together against other groups of generators.

Because power systems generally are large nonlinear time-varying systems, it is often difficult to dampen these oscillations. A static var compensator is one of several devices that can be used to enhance damping in a power system. Other devices are power system stabilizers (PSS), high voltage DC (HVDC) converters, and electronically controlled braking resistors.

Since a static var compensator (SVC) can modulate the power flow on the line where the controller is located, the controller can have a much greater effect on some of the interarea modes. Also an SVC may be more feasible to use for a utility company that is primarily a power transmission company and not a power generation company.

The objective of this thesis is to develop control strategies which can be applied to a static var compensator (SVC) in order to dampen low-frequency interarea oscillations in multimachine power systems. The primary purpose of most SVC's is local bus voltage control. An SVC acts effectively as a thyristor controlled variable shunt susceptance in the power network. In response to an input signal the shunt susceptance can be changed to modify the reactive power and voltage magnitude at the local bus. Interarea damping control can be added as a supplementary control loop using one of the system signals which has good controllability and observability characteristics for several interarea modes. It is expected that the design processes that are being developed will not be unique to SVC's but may also be applied to many other power system devices.

In this thesis two approaches to robust controller design are presented. Both of these approaches attempt to dampen low-frequency oscillations in a noisy multimachine power system environment.

The first approach is concerned with the ability of an advanced self-tuning adaptive controller to dampen low-frequency oscillations in large interconnected power systems when various levels of noise are present. The controller includes an on-line real-time recursive extended least-squares identifier to determine parameters of the system and the noise model; the model parameters are used in a real-time control-design algorithm to obtain digital controller parameters; and the resulting digital control algorithm uses both system outputs and past control actions to generate control signals for the process.

The second approach is a nonadaptive design technique for power system damping controllers. This nonadaptive technique is based on combined system identification and controller design. An iterative closed-loop identification method is used to find a linear model for the power system. To make the controller more robust to uncertainty, plant variations, and unmodeled dynamics, a linear quadratic Gaussian controller design method with loop transfer recovery (LQG/LTR) based on a generalized technique for the nonminimum phase (NMP) power system model is used to design the controller.

The simulations discussed in this thesis are based on the ETMSP software developed by the Electric Power Research Institute (EPRI) [1]. The extended/midterm stability program (ETMSP) is a time-domain simulation program for stability analysis of large power systems. In order to access ETMSP signals a subroutine had to be added to the program. The basic function of this subroutine is to allow access to any system signal within the ETMSP program. It also allows modification of the controller reference signals and variables. A manual is available for detailed information about the subroutine and state variables [2].

To evaluate the control design methods, they are simulated in realistic computer models of large power systems. Large power systems, such as the western grid of the United States, contain hundreds of generators and transmission lines. To achieve reasonable simulations in terms of time and cost, reduced-order systems are required that retain key attributes of the large systems. In Chapter 2 an approach to obtaining reduced-order power system test models is described. The approach has been developed and modified over the past few years at MSU. By using the approach, many essential features of large power systems are retained in significantly smaller power system models.

The remaining part of this chapter is organized into four sections. A brief review of SVC control for power system damping is given in the first section. Robust adaptive control for noisy power systems is introduced in section two, and robust nonadaptive control concepts are introduced in section three. In the last section the organization of the remaining thesis is outlined.

SVC Control for Power System Damping

The need to improve damping in power system networks has been growing over the years [3]. Poorly damped oscillations have been noticed in power systems in many parts of the world. The major factor contributing to these oscillations is often the use of long transmission lines carrying large power flows from one area of a system to another. In some cases this occurs because generating facilities are located in remote areas from major load centers. Undamped or poorly damped oscillations are often the limiting factor in determining how much power can be transported from one area of a system to another. Smith [4] gave a good review of damping controller designs using static var compensators which appeared in the literature prior to 1988. This section gives a brief review of some papers in this area which have been published recently.

Gyugyi [5] has written a paper about fundamentals of thyristor-controlled static var compensators in electric power system applications. The first part of the paper describes methods of reactive power generation, including power circuit arrangements, basic operating principles, and the internal control mechanism necessary to provide continuously variable var output. The second part explains the functional and operating requirements of power system compensation, with respect to voltage support and stability improvement, and derives the external controls needed to meet these requirements. The third part deals briefly with the coordination of the thyristor-controlled static var compensator and conventional mechanically switched capacitors and reactors.

In [6] Lerch, et al., describe a new method which defines the phase angles of generators on the basis of voltage and power measurements at the location of an SVC. These state variables are employed for improvement of damping of power system oscillations by the SVC. The new SVC control can optimally damp active poor oscillations. The damping signal is shown to be robust in particular in the vicinity of the stability limit and has the advantage that no error signals occur at large differences in the phase angles.

Padiyar, et al., [7] consider the application of a damping torque technique to examine the efficacy of various control signals for reactive power modulation of static var system (SVS) in enhancing the power transfer capability of long transmission lines.

Larsen, et al., [8] discuss the basic aspects of applying SVC's to series-compensated AC transmission lines which creates some new application considerations. The paper presents basic information on SVC interactions with series-compensated transmission lines. There exists a potential for adverse interactions which are different from the type seen in ac systems without series compensation. Secure operation can be attained by minor modification to the SVC control system, but specific studies are suggested - at least for each different SVC control type.

Dash, et al., [9] present an adaptive stabilizer design for static var compensator control in power systems, for either voltage regulation or controlling dynamic and transient performance under abnormal conditions such as 3-phase short-circuits, voltage collapse, load shedding, etc.

Hsu, et al., [10] examine the effect of a power system stabilizer and a static var compensator on the damping of a longitudinal power system. A combination of PSS and an SVC are employed because a large generating unit which is suitable for the installation of an additional PSS is not available in the central area of the study system. Results from time domain simulations indicate that the PSS and the SVC are very effective in damping system oscillations.

Robust Adaptive Control

Adaptive control has been an area of intensive research for the past thirty years (see [11] and [12] for example) and has reached the stage where the technology can be applied to produce significant benefits in certain applications. An area in which this appears to be true is in the control of large power systems [13]. The two main approaches to adaptive control are direct adaptive control (having at most implicit parameter identification) and indirect adaptive control (with explicit parameter identification). With indirect self-tuning control, an on-line real-time identifier determines parameters of a system model, the model parameters are used in a real-time control-design algorithm to obtain digital controller parameters, and the resulting digital control algorithm uses both system outputs and past control actions to generate control signals for the process.

In large interconnected systems that contain lightly damped modes of oscillations (e.g., power systems), controllers are often added to the system to increase the damping of certain modes. These controllers must respond to system disturbances in such a way that the control signal reacts to a disturbance, adding damping, and then returns to a nominal design-center value after the disturbance in order to be ready to react in either direction for the next major disturbance. Also, the control often must be limited in its rate of change in order to minimize the excitation of higher-order modes in the system. A control law that is ideal for the above purpose is an enhanced linear-quadratic (LQ) control law [14-16] that penalizes system states, control level, and rate of change of control level. Adaptive versions of this control are generalizations of the LQ adaptive controller described by Samson [17].

In [4] and [18], an adaptive version of enhanced LQ control was used in a nine-bus power system simulation to control a static var unit; the controller was single-input single-output, and a recursive least-squares (RLS) identifier was used. In [19] the controller was structured to add damping to the system in which the primary purpose of the static var unit was voltage regulation; two system outputs were used by the controller. The use of many such controllers as power system stabilizers on generator exciters in large interconnected power systems is examined in [20-23]; benefits are shown to accrue from the use of coupling terms in local identifiers when associated generators are closely coupled in the network. While the power system simulations used in the above studies included realistic power system dynamics (nonlinear models and system faults), the simulations did not include realistic levels of power system ambient noise.

In Chapter 3, a mass-spring system with abrupt parameter changes and additive system noise is used as a test bed for an enhanced LQ adaptive controller; it will be shown that recursive extended least squares (RELS) can be beneficial to adaptive damping by determining colored noise parameters in addition to system parameters [24].

The advanced adaptive controller designed in this way is applied in Chapter 4 to dampen low-frequency oscillations in a large interconnected power system simulation. Robustness of the control is examined relative to realistic load noise and faults that change the system operating point [25].

System Identification and Robust Nonadaptive Control

This part of the thesis is concerned with the use of least squares transfer function identification methods for the design of controllers to enhance the damping of interarea oscillations in multimachine power systems. A brief history of the early developments in transfer function identification in power systems can be found in [26]. Previous work in the area has shown that numerically identified transfer function models can be very useful in a variety of power system applications [27-31]. These applications include the tuning of damping controller parameters, software validation, model validation, and on-line evaluation of power system operating conditions in general. Recent papers in transfer function identification have reported significant advantages in identifying the plant from closed-loop operation rather than from the open-loop state [32-38].

In [32] the authors suggest that identification and model-based control design have to be treated as a joint problem if they are combined to achieve a high-performance control system. Solving this joint problem with individual identification and control design methods requires an iterative approach. The proposed iterative scheme is based on a robust control design method. Each identification step uses the previously designed controller to obtain new data from the plant. The associated identification problem has been solved by means of a coprime factorization of the unknown plant. An example has given evidence of the utility of the iterative scheme.

In [33] a criterion for system identification is developed which is consistent with the intended use of the fitted model for modern robust control synthesis. Specifically, a joint optimization problem is posed which simultaneously determines the plant model estimate and control design, so as to optimize robust performance over the set of plants consistent with a specified experimental data set.

In [34] a fractional representation approach is used to state and solve the closed-loop experiment design problem in terms of variables which are at the designer's disposal: the closed-loop inputs and the initial controller. The paper states that most actual identification experiments are conducted while the system is operating under closed-loop control and that the direct application of open-loop results to the closed-loop problem generally gives unsatisfactory results. This paper therefore addresses the problem of system identification experiment design in which the system to be identified is operating in a stable closed-loop configuration.

In [35,36] the authors suggest a combined iterative control system design which couples the separate stages of model identification (using frequency weighted least squares) with controller design from this model (using frequency weighted LQG methods).

Reference [37] focuses on the extension of an iterative identification/controller design using the H_2/LQG iteration of Zang, et al., [35,36]. The paper proposes two refinements to this iterative scheme. The first refinement is to decouple the identification and controller design phases by altering the sequence of steps in a single iteration. The first part of the iteration adjusts the model, while the second part refines the controller. The second refinement is a natural extension of the first. In this refinement they focus upon the controller enhancement with respect to the achieved and designed performance. This permits the consideration of an iterative controller design without the need for model adjustment.

Linear quadratic Gaussian controller design with loop transfer recovery (LQG/LTR) is an area of feedback control theory that has recently received a great deal of attention [39-41]. For systems that are nonminimum phase LQG/LTR controller design is based on partial loop transfer recovery techniques [41-43].

Organization of Thesis

The reminder of this thesis consists of seven chapters. Chapter 2 that follows describes an approach to obtaining reduced-order power system models that retain essential features of large power systems [31].

Chapters 3 and 4 are concerned with the robustness of a self-tuning adaptive controller to dampen low-frequency oscillations in large noisy power systems. In Chapter 3 a mass-spring system with abrupt parameter changes and additive system noise is used as a test bed for the design of an enhanced LQ adaptive controller. A recursive extended least-squares (RELS) algorithm is used for identification which determines colored noise parameters in addition to system parameters [24]. In Chapter 4 the advanced adaptive controller described in Chapter 3 is applied to dampen low-frequency oscillations in large interconnected power system simulations. The controller uses a static var unit to dampen low-frequency oscillations [25].

Combined closed-loop identification and controller design is considered as a nonadaptive design technique. Chapter 5 presents an introduction to concepts and applications of least squares transfer function identification (TFI) in power systems [29]. Chapter 6 shows the effectiveness of using closed-loop identified models compared to open-loop identified models for the design of power system controllers [44]. And Chapter 7 demonstrates the robustness of a generalized LQG/LTR controller design used for damping power system oscillations [45]; two approaches to partial LTR applicable to NMP plants are described and compared. The robustness of the resulting controllers is examined relative to the amount of loop transfer recovery that is possible for the NMP plant model.

Chapter 8 summarizes the contribution of this thesis and discusses some research topics that merit additional study.

CHAPTER 2

LOW-ORDER POWER SYSTEM MODEL

This chapter is concerned with the development and evaluation of a low-order power system transient stability test model which has dynamic characteristics similar to those exhibited by a full-order model [31]. The model is designed to have dynamic characteristics resembling those of a 2000 bus model of the western North American power system. The objective of the low-order modeling effort is to create an interesting and realistic environment in which to develop and test dynamic control strategies. Controller designs which appear promising on the test model can then be evaluated more thoroughly on the high-order system.

One of the major difficulties in the development of realistic dynamic control strategies for large interconnected power systems is the sheer size of many modern power networks. Detailed studies of such systems require substantial computational effort as well as the analysis of volumes of output data. The dynamic vastness precludes any attempt at controller design based on full analytic modeling. The study of power system dynamics and damping control therefore generally involves numerous simulations under a variety of system conditions. For a full-order model this is inconvenient due to both the computational time involved and the difficulty of sorting through the enormous amount of data that the simulations generate. Unfortunately, when the dynamic size is reduced to that which can be analytically described, the representation of the pertinent control environment may be lost. Thus the control strategies that are developed on a low-order system may not work well when applied to a full-order system due to large differences in the control environment of the two models. The most extreme example of a low-order model which cannot reasonably represent the dynamics of a large-order system is the one-machine infinite-bus system. Although the one-machine infinite-bus system has been very useful for developing an initial understanding of power system dynamics, the control environment is much too simplistic for many realistic studies intended for application to large-order systems. For these reasons there is a need for models which are reduced in complexity from a full-order model but reasonably reflect its characteristic dynamics.

This chapter outlines the development and characteristics of one low-order test model which has been developed and modified over the past few years at Montana State University. It has been used to design and test possible methods for increasing power system damping using several strategies. These strategies include PSS design [27], SVC modulation [29], and HVDC modulation [46]. The objectives are twofold: to describe the model development; and to illustrate techniques for the analysis of control system environments which may make designing controllers more difficult in some cases than in others.

The model under discussion was derived from a 2000 bus model of the western North American power system which is used by the Western Systems Coordinating Council. It has 46 buses, 19 machines and has power flow patterns and dynamic characteristics which are similar to the 2000 bus case. It was designed to have a similar enough control environment that experimental control strategies developed for the 46 bus model can be expected to transfer reasonably well to the 2000 bus model. A highly accurate reduced model is not the objective of this 46 bus system. The primary objective is a highly reduced model retaining a few local modes and the major interarea characteristics of the original system. The simulations discussed in this work have been carried out using the ETMSP software [1].

The organization of this chapter is as follows. First, the development of the low-order model is discussed, and the low-order model is compared to the high-order model using frequency domain plots of generator speed signals. Following this the Fourier transform is briefly examined in terms of transfer function residues and damping. Also some characteristics of the low-order model are examined and compared to the high-order model using a Prony based transfer function identification algorithm.

Low-Order Model Development

The frequency response of a generator state variable typically shows a number of oscillations in the 0.1 to 1 Hz range. These oscillations - called interarea modes - involve groups of generators swinging together against machines in another area of the system. The machine states also will usually exhibit local modes; that is oscillations in the 1 to 3 Hz range which are due to the machine itself swinging against the system. Because the interarea modes involve groups of machines swinging together, they lend themselves well to being modeled in a reduced-order environment where a number of machines are represented by a single big machine. The local modes, on the other hand, are each due to the local machine interactions and so as the number of the machines being individually modeled is reduced, much of the detail of the local dynamics may be lost.

A reasonable goal, therefore, in constructing a low-order model is to reproduce the interarea interactions as well as possible and to create a single local mode on each of the machines which is similar to the modes which a machine in the high-order system exhibits. A description of the steps in constructing this low-order equivalent are listed below. Each of these steps is discussed in more detail in the subsections which follow.

1. Geographic - Each major area in the system is represented by at least two machines, one classical and one or more detailed. The areas are then connected by intertie lines in a similar geometry to the 2000 bus system.

2. Powerflow - The power flows between areas are adjusted to match those of the 2000 bus case.

3. Kinetic Energy (KE) - The KE (MWatt-secs) of the classical machines are adjusted to move interarea modes until they line up well with the 2000 bus case.

4. Damping Coefficient - The damping coefficient on each machine is adjusted until the damping of the local mode is similar to that of the relevant machine in the high-order system.

Geographic

Figures 2.1 and 2.2 are both representations of the western North American power system. Figure 2.1 shows the actual layout of the major transmission lines.

Figure 2.1. Major transmission lines of the western North American power network.

Figure 2.2 is taken from a Western Systems Coordinating Council (WSCC) bubble diagram description of a planning case and shows the ownership regions (areas) of the system and power exchange over interties. The first step in constructing the low-order model involves consolidating some of the ownership areas. This has been done for this case largely by experience gained from running numerous large-order transient stability simulations as well as from system field tests [47,48]. This knowledge of major interarea modes allows grouping of regions which usually move together. In general if the dynamic groupings are not already known, they can be determined from eigenanalysis as in [49]. Each of these groups, which is surrounded by dashed lines in Figure 2.2, is an area in the low-order model and is represented by a 500 KV bus or a 345 KV bus in the case of UTAH.

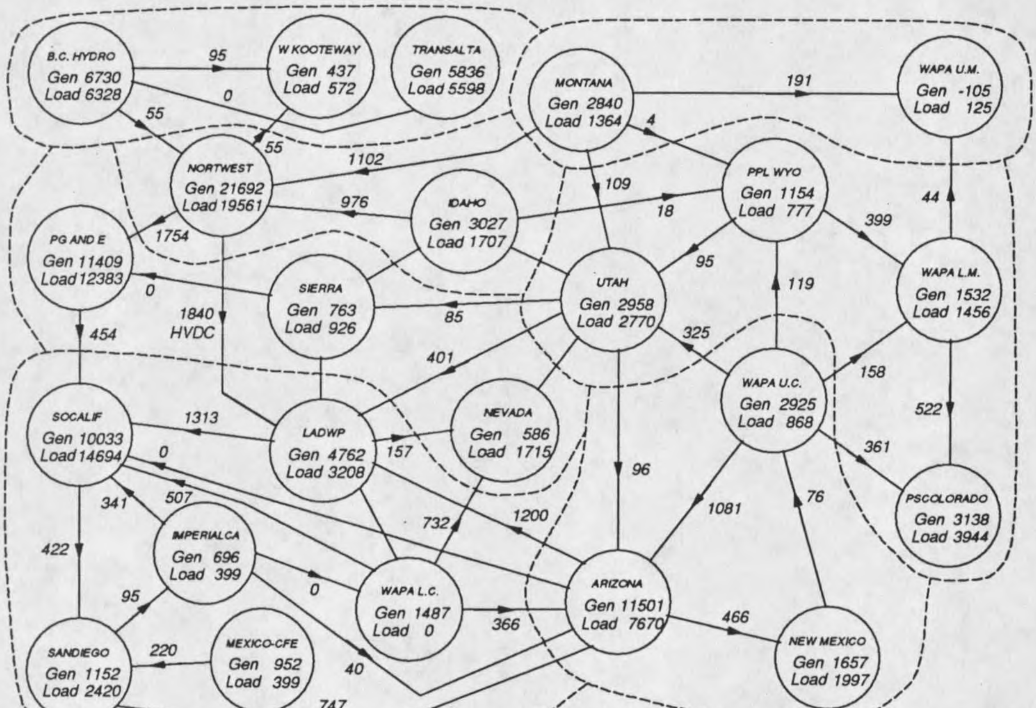

Figure 2.2. WSCC power flow interchange diagram.

At least two generators are placed in each of the areas in the low-order model. One is a very large power and high KE machine represented by classical swing equations and a constant voltage behind a machine reactance. This type of machine, which will be referred to as an Inertial Composite Machine (ICM), is intended to represent a conglomerate of smaller machines found in the full-order case. Note that although typically the data field for a classical machine model requires kinetic energy of the machine at 60 Hz (in either MW-s or MW-s/MVA), there is a linear relationship between this quantity and the moment of inertia of the machine. The large generating power and large KE of the composite machines is what creates the interarea dynamics in the low-order system. The ICM also provides the bulk of generation in the area. The MVA base of the ICM is chosen to be sufficiently large to provide the necessary generation. The machine reactances for the ICMs are entered as typical values in per unit which will then be scaled up by the large MVA base. The choice of the KE for the ICMs is discussed in a following sub-section. The other machines in each area are full d-q axis generator models [50,51]. These machines approximate actual machines in the area, and the data for them is derived from actual machines. For example, in the low-order model of Figure 2.3 there are two full machines connected to the bus NORTH. NORTH_G1 is a machine similar to MCNARY and NORTH_G2 is similar to CHIEF JO, where MCNARY and CHIEF JO are actual machines in the western system. The detailed machines in this model are often representations of a number of small machines at a single generating station lumped into one model. The ICMs are indicated in Figure 2.3 by the larger circles and labeled as CM. In the data file the ICMs have names ending in CM such as NORTH_CM or ARIZO_CM.

The last step in the geometric construction of the low-order system involves connecting the regions in the low-order system together appropriately by interties. This is done by examining Figure 2.2 and providing similar interties to those that exist in the actual system. The transmission line data is estimated by using typical values of transmission line impedance per distance with approximate accounting of distances between different areas and the number of parallel connections. The 2000 bus model being used as the basis for the low-order model has two HVDC lines. These are the Pacific and the Intermountain HVDC transmission lines. For the purposes of this version of the 46 bus model the DC lines are being represented as constant current loads in both the high-order and low-order cases. A detailed HVDC model has been developed for both the four terminal Pacific line and the two terminal Intermountain line using the DCMP features of the ETMSP program [1]. These HVDC models are very complex and difficult to represent in many stability programs and so they have been omitted from the studies discussed here. The possibility of a simplified but accurate DC representation, especially for the Pacific HVDC system, should be considered.

Power flow

The next step is to obtain an initial convergent power flow solution to the low-order model which has similar intertie power flows to that of the 2000 bus case. This is usually easily accomplished if the power flow program allows area interchange to be specified. If this is the case then the area interchanges can be specified and the power flow program allowed to adjust the generation on each of the ICM's until convergence is obtained. If the power flow program does not allow specific interchanges then this step can require a good deal of tinkering with power flow parameters. A diagram of the low-order model with power flows is shown in Figure 2.3.

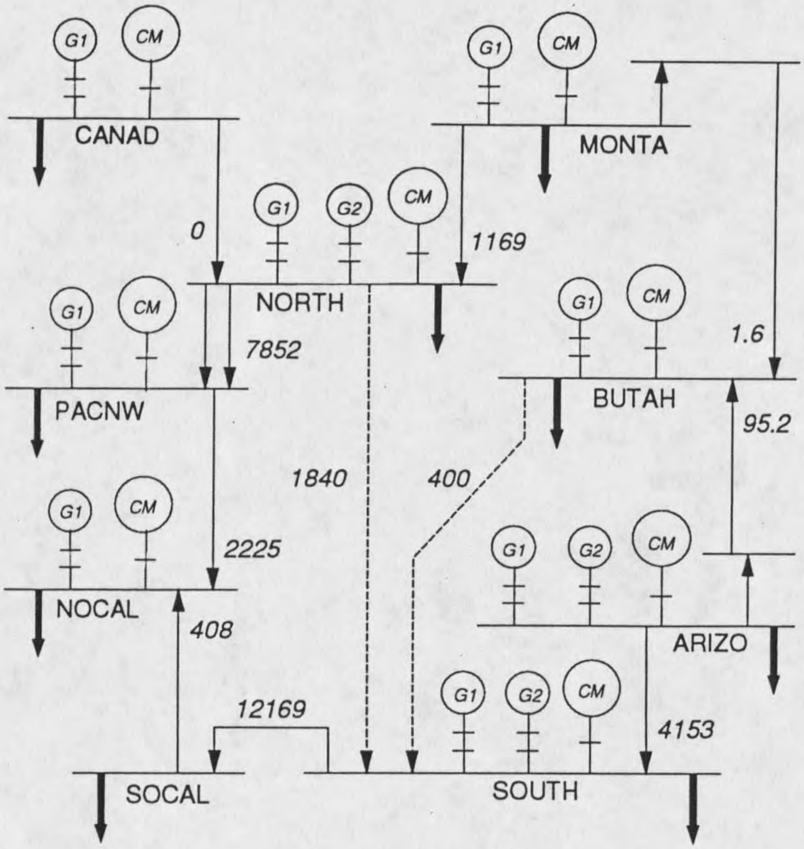

Figure 2.3. A low-order model of the western North American power system.

Once the initial power flow case converges then shunt reactances at the major buses are adjusted to approximately match voltage levels in the full-order case. Line impedance estimates between areas are then refined by adjusting the values in an iterative fashion and referring to the full-order solution to make the difference in voltage angles between the various areas match. For example, the full-order power flow solution may show voltage angles on centrally located 500 kV buses in two different areas to have a difference of about 20 degrees. The impedance of low-order model transmission lines are adjusted until the solution shows the same voltage angle difference for the same power transfer between areas. In this way the effects of numerous parallel connections, series line compensation and inexact distances between areas are taken into account to produce a model that is electrically similar.

Kinetic Energy

An initial value for the KE on the ICMs is established by scanning the full-order stability case data file to find the average KE per generating power for machines within that area. The KE for each ICM is then estimated by scaling this average value up to the ICM's MVA base.

To further adjust the ICM KEs it is necessary to begin comparing the results of transient stability simulations of the low-order model and the high-order model. The exciter reference voltage is pulsed for each of the full machine models in the low-order case and for each of the associated machines in the 2000 bus case. The resulting speed and accelerating power signals are then transformed into the frequency domain using the discrete Fourier transform via a Fast Fourier Transform algorithm (FFT). A typical example of a low-order versus high-order speed signal before the adjustment of ICM KEs is shown in Figure 2.4. Notice that the modes are similar but the frequencies do not line up particularly well. By adjusting the KEs on the ICMs the interarea modes can be moved until they match up as shown in Figure 2.5.

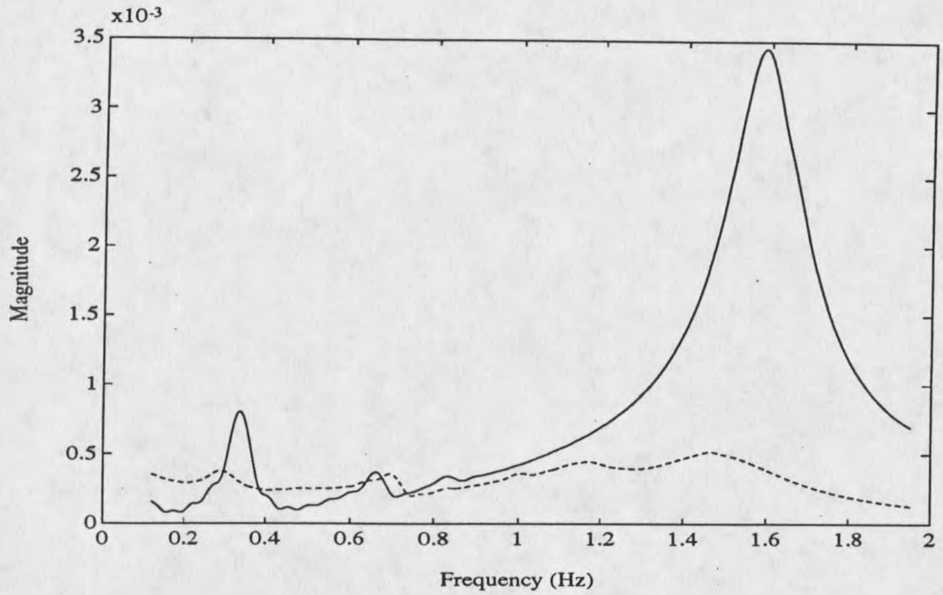

Figure 2.4. A comparison of high-order (dashed) and low-order (solid) speed signals in the frequency domain before any adjustment of inertia or damping.

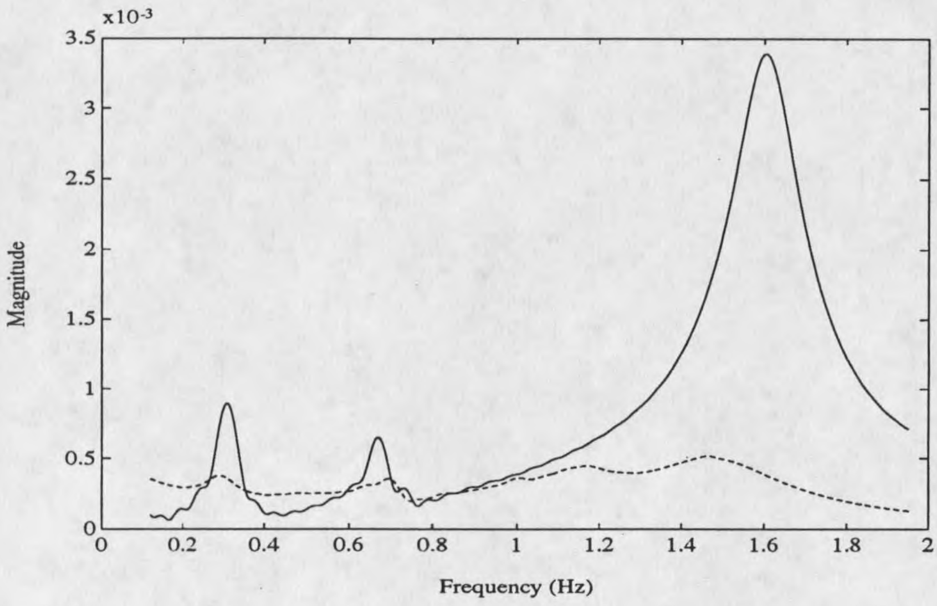

Figure 2.5. A comparison of high-order (dashed) and low-order (solid) speed signals in the frequency domain after KE adjustment.

The use of eigenanalysis [52,53] has proven to be very helpful in this process by identifying which machines are participating the most heavily in each of the interarea modes. This information helps to clearly identify which ICM KE needs to be adjusted to move a specific mode. Table 2.1 lists the machines with the largest participation factors in each of the interarea modes for the low-order model. It is interesting to note that the ICMs have the largest influence in each of the interarea modes listed. As KE is increased for a machine participating in

Mode	High Participations
0.31 Hz	ARIZO_CM [1.0], CANAD_CM [0.6], NORTH_CM [0.4], SOUTH_CM [0.3]
0.56 Hz	BUTAH_CM [1.0], CANAD_CM [0.4], BUTAH_G1 [0.3]
0.61 Hz	CANAD_CM [1.0], NOCAL_CM [0.97], ARIZO_CM [0.5], SOUTH_CM [0.3]
0.71 Hz	NORTH_CM [1.0], BUTAH_CM [0.7], SOUTH_CM [0.6], ARIZO_CM [0.6],CANAD_CM [0.5]
0.83 Hz	NOCAL_CM [1.0], SOUTH_CM [0.4], NOCAL_G1 [0.2], SOCAL_G1 [0.2]

Table 2.1. A list of the machines that participate highly in some of the major interarea modes for the low-order model with normalized participation factors in brackets.

a mode, the frequency of the mode tends to decrease. Thus by examining Table 2.1 it can be concluded that the 0.31 Hz mode can be decreased in frequency by increasing the KE on ARIZO_CM. This would also have a minor effect on the 0.61

Hz mode. To adjust the 0.71 Hz mode one would select the NORTH_CM machine and so on. These relations have been verified by time domain simulation and FFT analysis as illustrated in Figure 2.5.

Damping Coefficients

The last parameters to be adjusted are the damping coefficients of the detailed machines. As can be seen in Figure 2.5, the local mode is relatively large on the low-order model (ARIZO_G1). This is because the low-order model does not yet contain any power system stabilizer (PSS) units but the machines in the 2000 bus model do (Palo Verde in this case). If the damping coefficient on a detailed machine in the low-order model is increased, this has a similar effect to inserting a properly tuned PSS unit, and the local mode of the machine is decreased. An FFT of the machine speed signal after adjustment of the machine damping coefficient is shown in Figure 2.6. In contrast, Figure 2.5 illustrates the same signal when the damping coefficient is zero. Figures 2.4, 2.5, and 2.6, illustrate a progression of improvements in the frequency response of a low-order signal relative to the high-order system as the KEs and damping coefficients are adjusted in the low-order system.

Dynamic Characteristics

In the previous section data from the low-order model is compared to data from the high-order model using frequency domain plots which are generated using an FFT algorithm. Frequency domain plots are used because they are a convenient, graphical, and quick way to look at the behavior of the system as the parameters are being adjusted. The purpose of the low-order model is to act as a test bed for control strategies and so it is necessary that the control environment of the low-order model be similar to that of the high-order model.

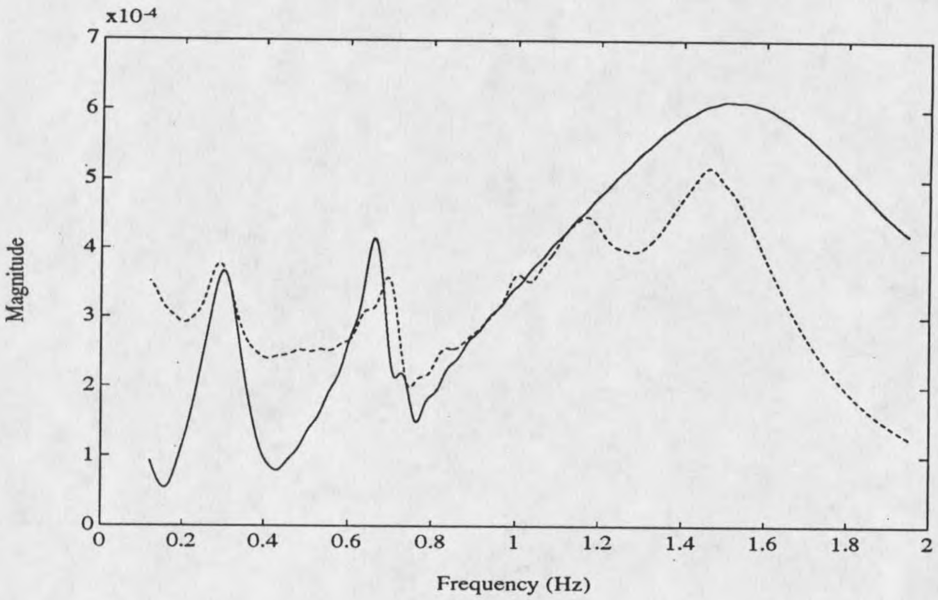

Figure 2.6. A comparison of high-order (dashed) and low-order (solid) speed signals in the frequency domain after KE and damping coefficient adjustment.

Lightly damped oscillatory modes within a signal are characterized by peaks in the FFT plot of the signal. One way to establish similarity between the control environments is to compare the peaks such as those exhibited in Figure 2.6. However, frequency response plots as generated from an FFT algorithm do not explicitly reveal residue and damping information about a particular mode. To obtain more explicit information such as the residue and damping terms associated with a particular mode it is necessary to obtain transfer functions between the input signal and the observed system response signal. The results of one algorithm for transfer function identification are discussed in a following subsection but first the Fourier transform results are discussed to establish what information can be obtained from the frequency domain plots and how it is related to transfer function residues and modal damping terms.

Figure 2.7 illustrates a system with frequency response $T(j\omega)$. $y(t)$ is the time domain output which is transformed into the frequency domain signal $Y(j\omega)$ using an FFT. The FFT is related to the Fourier transform and so the frequency domain signals are related by

$$Y(j\omega) \approx T(j\omega)X(j\omega)\,. \tag{2.1}$$

If the input signal $x(t)$ is a pulse of sufficiently short duration, so that $X(j\omega)$ is flat in the frequency range of interest, then $Y(j\omega)$ is a good approximation to a scaled version of the frequency response of the system. Further justification for the validity of referring to Figures 2.4-2.6 as frequency responses is displayed in Figure 2.8 which compares the frequency domain signal from Figure 2.6 with a frequency response generated by the Epri's SSSP eigenanalysis program [53]. SSSP generates the frequency response using the eigenvalues of a state space system description and does not do a time domain simulation at all. Thus the two curves in Figure 2.8 are generated in entirely different manners. In the FFT plot the local mode is reduced from what it actually should be due to the frequency domain characteristics of the input being less than perfectly flat.

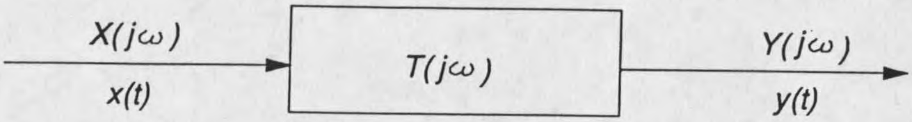

Figure 2.7. Block diagram illustrating input and output signal relations.

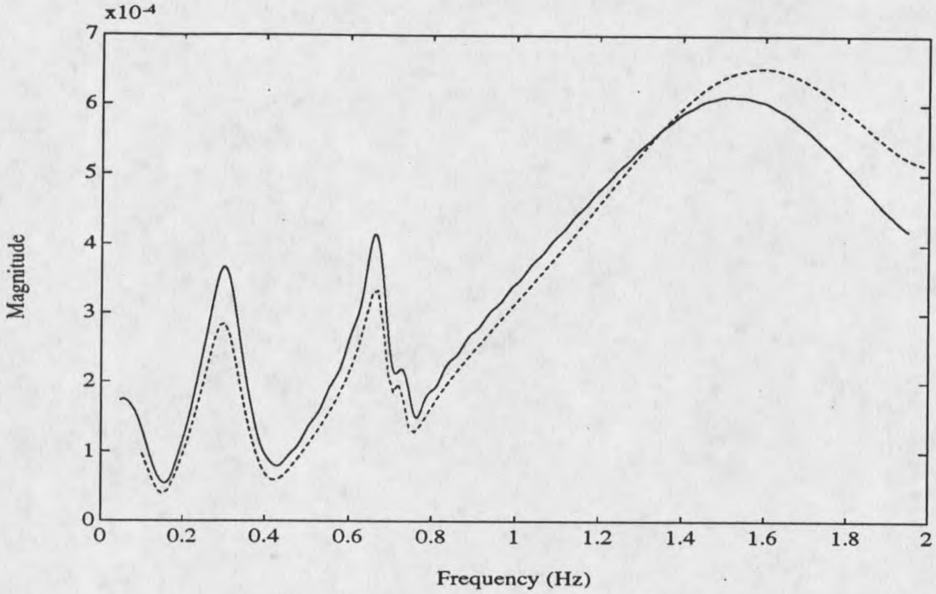

Figure 2.8. A comparison of frequency domain plot of speed signals (solid) and frequency response generated using SSSP (dashed).

The next step in examining the FFT results is to establish what the magnitudes of the peaks in the FFT plot mean. Power system response signals typically contain several damped sinusoidal components associated with different interarea modes. For the sake of clarity we take for an example, a signal composed of a single damped sinusoid and examine it analytically using the Fourier transform. The time domain description of such a signal can be expressed as:

$$y(t)=\left[R_1 e^{(\sigma_1+j\omega_1)t} + \overline{R}_1 e^{(\sigma_1-j\omega_1)t} \right] u_s(t) \tag{2.2}$$

where $u_s(t)$ is the unit step function and R_1 is the signal component residue and the bar indicates its complex conjugate. The damping term σ_1 is the real part of the eigenvalue and the frequency ω_1 is the imaginary part. The Fourier transform of this signal is

$$Y(j\omega)=\int_0^\infty e^{\sigma_1 t}\left\{ R_1 e^{j(\omega_1-\omega)t} + \overline{R}_1 e^{-j(\omega_1+\omega)t} \right\} dt\,. \tag{2.3}$$

The magnitude of the Fourier transform of this signal evaluated at $\omega=\omega_1$ is then given by

$$|\,Y(j\omega_1)\,|=|\frac{R_1}{\sigma_1}+\frac{\overline{R}_1}{\sigma_1-2j\omega_1}|\,. \tag{2.4}$$

For many power system signals containing lightly damped interarea modes, where σ_1 and ω_1 are expressed in radian quantities, it turns out that $2\omega_1 >> \sigma_1$. With this assumption we can make the following approximation:

$$|\,Y(j\omega_1)\,|\approx|\frac{R_1}{\sigma_1}+\frac{\overline{R}_1}{-2j\omega_1}|=|\frac{-2j\omega_1 R_1+\overline{R}_1\sigma_1}{-2j\omega_1\sigma_1}|\,. \tag{2.5}$$

And, again, using $2\omega_1 >> \sigma_1$,

$$|\,Y(j\omega_1)\,| \approx |\frac{-2j\omega_1 R_1}{-2j\omega_1\sigma_1}| = |\frac{R_1}{\sigma_1}|\,. \tag{2.6}$$

Thus the peak magnitude of the Fourier transform at the modal frequency for lightly damped modes is approximately equal to the magnitude of the residue of the mode divided by the damping term of the mode. Most electromechanical modes are in the range from 0.62 to 12.6 (rad/sec); therefore, the approximation is fairly good for damping values around 0.1 nepers or less. As the frequency of the mode increases this approximation improves.

Transfer Function Identification

One way to obtain residue and damping terms for each mode is to use transfer function identification methods [28,48 and 54]. In the applications illustrated here transfer function identification methods based on the Prony algorithm are employed. Using transfer function identification to compare data generated by the low-order model and the high-order model, a number of observations are possible. The first of these involves the similarity of the control environments and is illustrated in Table 2.2. This table shows the results of Prony analysis done on the same output data which was used to generate Figure 2.6. The comparison is made with a machine similarly located in the high-order model (ARIZO_G1 in the low-order model and Palo Verde in the 2000 bus case).

The results displayed in Table 2.2 indicate a good similarity in the interarea modes between the low and high-order models. The damping terms in the low-order model are both within 20% of the high-order model and if the residues on the high-order model are all scaled by 22, then the low-order residues are within 10%. The scale factor of 22 comes from the ratio of the exciter gains on the machines being examined in the low-order and high-order models. In the low-order case the exciter gain is set at 150 and in the high-order case the exciter gain is set at 6.8. The local mode on the low-order machine does not agree very well with the main local mode on the high-order machine. This is not surprising. By representing many machines with a single ICM it is not possible to retain the complexity of local behavior that exists in a high-order environment. In the high-order model there are a number of modes above 1 Hz whereas the low-order model exhibits only one. This is reflected in Figure 2.6 which clearly illustrates that the low-order model has a simpler frequency domain structure above 1 Hz.

Low-order			High-order		
Mode (Hz)	Damping (Nepers)	Residue (mag)	Mode (Hz)	Damping (Nepers)	Residue (magx22)
0.307	-0.301	2.304	0.294	-0.273	2.288
0.677	-0.174	1.100	0.716	-0.147	1.188
1.55	-2.87	59.4	1.49	-1.060	30.39

Table 2.2. A comparison of residue and damping terms in low and high-order models generated by doing Prony analysis on the speed signal after an exciter change.

Load Model Change

The low-order model is formulated with constant current load models. Recent results reported in [55] indicate that for a two area test system, constant impedance load models yield better damping of a 0.3 Hz interarea mode than constant current and constant power load models do. The low-order model under discussion was reformulated with constant impedance and constant power loads to see if a similar effect occurs. Figure 2.9 shows a frequency domain plot of the speed signal before and after changing the load models in the low-order model from constant current to constant impedance.

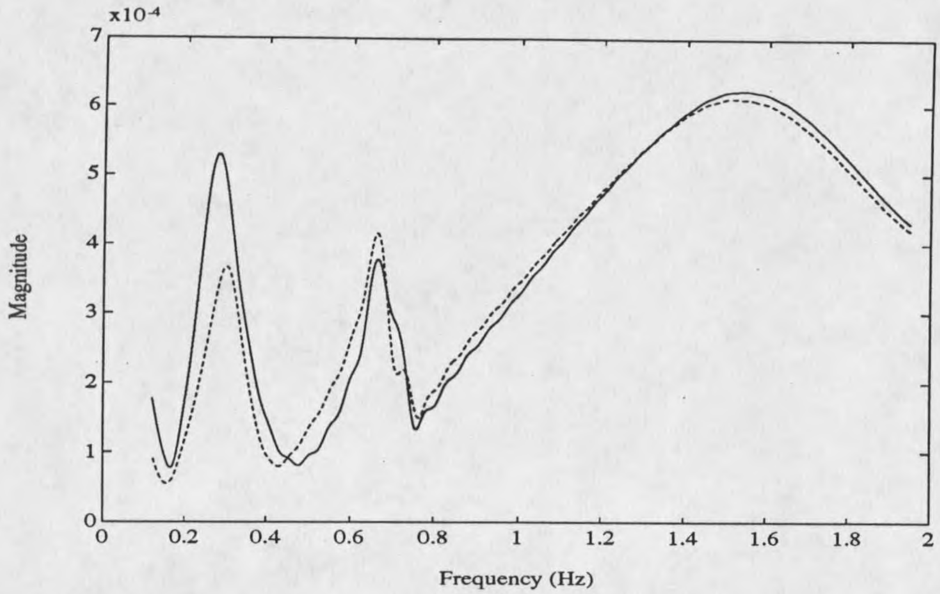

Figure 2.9. A comparison of frequency domain plots with constant impedance (solid) and constant current (dashed) loads.

The frequency domain plot indicates greater activity of the 0.3 Hz mode for constant impedance loads relative to constant current loads. Prony results displayed in Table 2.3 indicate that the damping at 0.3 Hz is increased (which would decrease the peak size) but the residue is increased by a larger percentage thus causing a larger peak at 0.3 Hz. The 0.67 Hz mode shows a slight reduction in magnitude in the frequency domain when the load models are changed from constant current to constant impedance and in this case Prony indicates a reduction in both damping and residue. Also included in Table 2.3 are the results of mode changes when the system load models are changed to constant power. As the load models are changed from constant impedance to constant current to constant power the damping of the 0.3 Hz mode becomes less and the ability to control the mode also becomes less as indicated by a smaller residue. By contrast, the damping of the 0.6 Hz mode becomes greater and the residue becomes greater also, indicating a greater sensitivity to feedback. These dynamic effects due to load model changes are consistent with those reported in [55] in that the constant current load model effects are intermediate between the other two. For both modes the changes in the magnitudes of the peak in the frequency response plots are in the same direction as the damping change rather than opposite to it. This is because in both cases the change in the residue term dominates over the change in the damping. This example illustrates the difficulties of trying to evaluate changes in system damping from frequency response plots.

Load Model	Mode (Hz)	Damping (nepers)	Residue (mag.)
Constant	0.277	-0.317	3.900
Impedance	0.667	-0.142	0.750
Constant	0.307	-0.301	2.304
Current	0.677	-0.174	1.100
Constant	0.352	-0.298	0.905
Power	0.661	-0.211	1.317

Table 2.3. Effects on the modes when load models are changed.

Discussion of Results

The development and characteristics of a low-order transient stability model of the western North American power system have been presented. The procedure for this model development is conceptually simple and yet the resulting system is quite adequate for developing and testing controllers for interarea damping. The low-order model is appropriate for use primarily in initial development of control strategies where the more accurate details of a full-order model are not absolutely necessary. Control strategies which appear to be successful on this low-order system model can be evaluated more extensively by simulations on full-order system models.

The Fourier transform was briefly examined in terms of transfer function residues and damping. Due to its speed and the graphical nature of its output the FFT serves as a useful tool in developing the low-order model. For detailed information about residues and damping, it is necessary to use transfer function methods to compare the control environments of the low and high-order systems. Fourier transform results were found to compare well to a frequency response generated using SSSP. It was found that changing the load models from constant impedance, to constant current, and again to constant power loads, has differing effects on the 0.3 Hz mode and the 0.67 Hz mode indicating that there is not a simple, predictable effect on system dynamics due to load model changes.

A possible extension of this model would include intermediate levels of modeling between the 46 bus system and the 2000 bus system. Representing one or more areas by detailed models rather than composite machines and interconnecting them to areas modeled as described will result in stability models which are suitable for studies where both local and interarea modes are important. The absence of any detailed modeling in any of the areas of the low-order model means that it may not do a good job of reproducing the local dynamic environment. Control strategies applied to PSS units, SVC's, HVDC converters, or other FACTS devices in this test model will find a control environment that is more similar to an actual large power system model than many simplified models may provide.

CHAPTER 3

THE ROBUSTNESS OF AN ADVANCED ADAPTIVE CONTROLLER AGAINST SYSTEM NOISE

This chapter examines the robustness of an advanced adaptive controller against system noise and abrupt parameter changes. Recursive extended least squares (RELS) is incorporated in a self-tuning adaptive controller which uses an enhanced linear-quadratic control algorithm. Simulations are given to demonstrate the effectiveness of the controller in a noisy environment. The system that is being controlled is a piecewise linear system with an independent noise input and with time-scheduled parameter changes unknown by the controller. Effects of noise level are examined along with trade-offs that must be considered regarding probing-signal strength and noise level. The robustness of this self-tuning control is shown to benefit from the use of RELS in place of RLS.

In this chapter first, the controller structure is examined: the equations for an RELS identifier are given; observer equations that conform to the RELS identified model are obtained; and the enhanced linear quadratic controller equations are presented for the case of two system outputs and one control input. Following this a particular electromechanical test system is described. The system contains lightly damped oscillatory modes, parameters that can change abruptly, and an independent noise input. The chapter ends with simulation results and concluding remarks.

Controller Structure

A self-tuning adaptive controller based on an enhanced LQ control algorithm is used as the controller. The adaptive controller has three components: 1) an RLS or an RELS identifier to identify coefficients of a difference equation model of the system being controlled; 2) an adaptive observer for obtaining state estimates of the system; and 3) an adaptive LQ controller. A control system diagram is illustrated in Figure 3.1 for the case in which the system has two outputs y_1 and y_2, a control input u, and a disturbance input d that is not directly measurable. The identified parameters are sent along with the latest values of the control signal u and outputs y_1 and y_2 to the adaptive observer and the enhanced LQ controller.

Recursive On-Line Identifier

Many different on-line identification methods have been developed. Among these methods, those based on recursive least-squares (RLS) are the most popular. The method called recursive extended least squares (RELS) is particularly useful for obtaining system models in the presence of colored noise.

A general representation of a single-input single-output linear system in the presence of a disturbance is given by

$$Y(z) = G_1(z)U(z) + V(z) \quad \text{and} \quad V(z) = G_2(z)E(z)$$

or

$$Y(z) = G_1(z)U(z) + G_2(z)E(z) \tag{3.1}$$

where u is the plant input variable, y is the plant output variable, and $\varepsilon = Z^{-1}\{E\}$ represents the disturbance input. In this work the transfer functions G_1 and G_2 are characterized as follows:

$$G_1(z) = \frac{b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{n_b} z^{-n_b}}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{n_a} z^{-n_a}} \qquad (3.2a)$$

and

$$G_2(z) = \frac{1 + c_1 z^{-1} + c_2 z^{-2} + \cdots + c_{n_c} z^{-n_c}}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{n_a} z^{-n_a}} \qquad (3.2b)$$

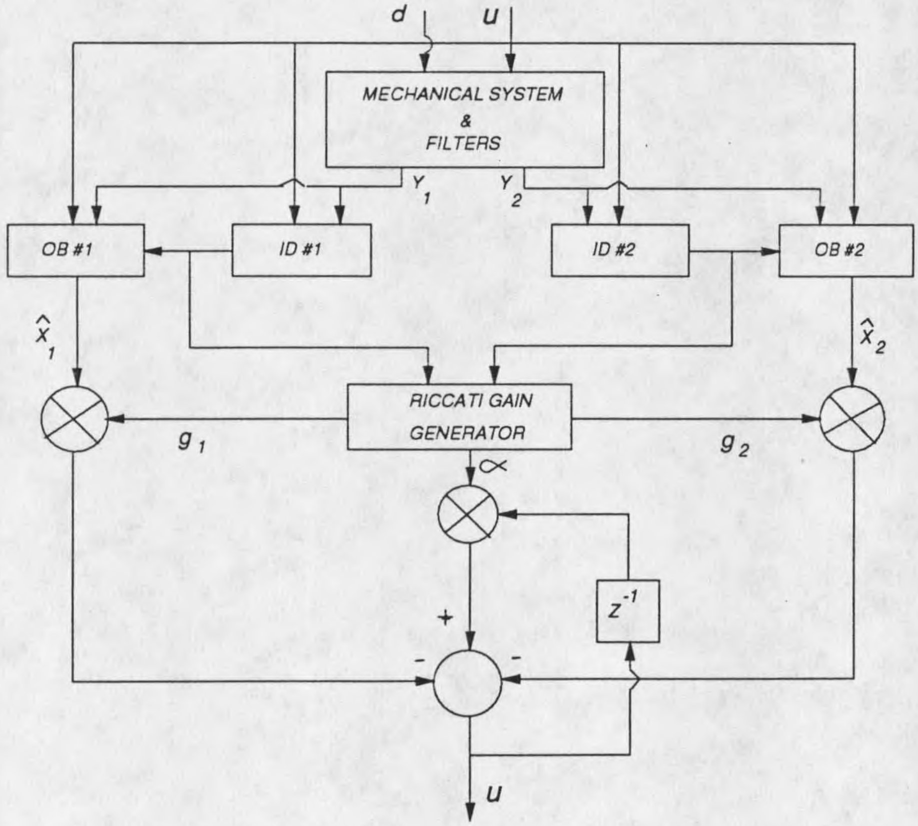

Figure 3.1. Block diagram for enhanced LQ adaptive control.

The primary objective of the identification procedure is to identify or estimate the unknown parameters in (3.1) and (3.2) by operating on sampled values of the input u and the output y. A secondary objective is to identify the error term $\varepsilon(k) = y(k) - y^*(k)$ which is the difference between the system output and the identifier's estimated output.

Equations (3.1) and (3.2) can be written in the following form:

$$\begin{aligned} y(k) = &- a_1 y(k-1) - a_2 y(k-2) - \cdots - a_{n_a} y(k-n_a) \\ &+ b_1 u(k-1) + b_2 u(k-2) + \cdots + b_{n_b} u(k-n_b) \\ &+ \varepsilon(k) + c_1 \varepsilon(k-1) + c_2 \varepsilon(k-2) + \cdots + c_{n_c} \varepsilon(k-n_c) \end{aligned}$$

or

$$y(k) = \phi^T(k)\pi(k) + \varepsilon(k) \tag{3.3}$$

where

$$\phi^T(k) \equiv [a_1 \quad \cdots \quad a_{n_a} \quad b_1 \quad \cdots \quad b_{n_b} \quad c_1 \quad \cdots \quad c_{n_c}] \tag{3.4a}$$

and

$$\begin{aligned} \pi(k) \equiv [&-y(k-1) \quad \cdots \quad -y(k-n_a) \quad u(k-1) \quad u(k-2) \quad \cdots \\ &u(k-n_b) \quad \varepsilon(k-1) \quad \cdots \quad \varepsilon(k-n_c)]^T \end{aligned} \tag{3.4b}$$

Equation (3.3) is known as a regression equation, and the elements of ϕ are regression coefficients. In the case that c_i's are not included in ϕ, RLS is applicable; if c_i's are included, RELS is applicable. Variations of RLS and RELS are used to minimize a weighted sum of squared errors and may include a forgetting factor [20]. The basic parameter update equation is

$$\phi(k) = \phi(k-1) + K(k)\,[y(k) - \phi^T(k-1)\pi(k)] \tag{3.5}$$

where $K(k)$ is the estimator gain vector and can be calculated by different methods. One of the methods for updating $K(k)$ is UD factorization which is numerically robust [56].

An estimated value $y^*(k)$ of $y(k)$ is given by

$$y^*(k) = \phi^T(k)\pi(k)$$

and the $\varepsilon(k)$ of (3.3) can be expressed as $y(k) - y^*(k)$. An objective in selecting the regression coefficients (the parameter vector) in (3.3) is to make $\varepsilon^2(k)$ small.

Consider the system characterized by $y(k) = \phi^T(k)\pi(k)$, with $\phi(k)$ and $\pi(k)$ given in (3.4). With $n_a = n$, $n_b \le n$, and $n_c \le n$, an equivalent state space description of the system is

$$x(k) = Ax(k-1) + Bu(k-1) + C\varepsilon(k-1) \tag{3.6}$$

and

$$y(k) = C_o x(k) \tag{3.7}$$

where

$$A = \begin{bmatrix} -a_1 & 1 & 0 & \cdots & 0 \\ -a_2 & 0 & 1 & \cdots & . \\ . & . & . & \cdots & . \\ . & . & . & \cdots & . \\ . & . & . & \cdots & . \\ -a_{n-1} & 0 & . & \cdots & 1 \\ -a_n & 0 & . & \cdots & 0 \end{bmatrix} \tag{3.8}$$

$$\begin{cases} B^T = [b_1 \quad b_2 \quad \cdots \quad b_{n_b} \quad 0 \quad 0 \quad \cdots \quad 0] \\ C^T = [c_1 \quad c_2 \quad \cdots \quad c_{n_c} \quad 0 \quad 0 \quad \cdots \quad 0] \\ C_o = [1 \quad 0 \quad \cdots \quad 0] \end{cases} \tag{3.9}$$

This state-space description is used in the formation of the observer.

Adaptive Observer

An observer estimates or reconstructs the unmeasurable states of a system. In this case, the observer uses the model of the system which is produced by the identifier. The estimated states are used in place of the actual states to form the control signal $u(k)$. Based on the system model of (3.6) and (3.7), the equations for the estimated states are

$$\hat{x}(k+1) = A\hat{x}(k) + Bu(k) + C\varepsilon(k) + K_e(y(k) - \hat{y}(k)) \tag{3.10}$$

where K_e is the observer feedback gain. Using $\hat{y}(k) = C_o\hat{x}(k)$, (3.10) is equivalent to

$$\hat{x}(k+1) = (A - K_eC_o)\hat{x}(k) + Bu(k) + C\varepsilon(k) + K_e y(k) \tag{3.11}$$

The objective of the observer is to force $y(k) - \hat{y}(k)$ to zero. One possibility is to place the observer poles at 0 in the z plane by using

$$K_e = [-a_1 \quad -a_2 \quad \cdots \quad -a_n]^T$$

Control Calculations

The objective of the adaptive controller of Figure 3.1 is to minimize the following performance measure J:

$$J = \sum_k \{y_1^2(k) + y_2^2(k) + w_1 u^2(k) + w_2[u(k) - u(k-1)]^2\} \tag{3.12}$$

where the summation is over future sampling instants, w_1 and w_2 are the weights on input and the rate of change of input, and y_1 and y_2 are two output signals of interest.

By considering the matrices A_1, A_2, B_1, B_2, C_1, and C_2 obtained from two identified transfer function models, one for each output, a combined state space system model is

$$x(k+1)=\begin{bmatrix} x_1(k+1) \\ x_2(k+2) \end{bmatrix}=Ax(k)+Bu(k)+C\varepsilon(k) \tag{3.13}$$

and

$$y(k) \;=\; C_o x(k) \tag{3.14}$$

where

$$A=\begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix},\quad B=\begin{bmatrix} B_1 \\ B_2 \end{bmatrix},\quad C=\begin{bmatrix} C_1 \\ C_2 \end{bmatrix} \tag{3.15}$$

and

$$C_o=\begin{bmatrix} 1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \end{bmatrix} \tag{3.16}$$

Using the A, B, C, and C_o as above, the observer states are characterized by (3.10) with $K_e=\begin{bmatrix} K_{e1}^T & K_{e2}^T \end{bmatrix}^T$ and $y(k)-\hat{y}(k)=[y_1(k)-\hat{y}_1(k) \quad y_2(k)-\hat{y}_2(k)]^T$.

The optimal control signal for minimizing J in (3.12) is given by

$$u(k)=\alpha u(k-1)-g_1^T\hat{x}_1(k)-g_2^T\hat{x}_2(k) \tag{3.17}$$

Ideally, the optimal gains α, g_1, and g_2 are obtained from the steady-state solution of an appropriate set of Riccati equations [15,21]. However for this adaptive control, because of the time-varying estimates of system parameters, the following Riccati equations are solved iteratively with only one iteration of the equations per sample period.

Initial values needed to start the iteration are

$$s = w_1 \frac{w_2}{w_1 + w_2} \tag{3.18}$$

$$Q_1 = Q_2 = [0 \quad 0 \quad \cdots \quad 0]^T \quad (nx1 \text{ vectors}) \tag{3.19}$$

$$R_{11} = R_{22} = [\delta_{i-1,j-1}]_{nxn}, \quad \text{and} \quad R_{12} = [0]_{nxn} \tag{3.20}$$

where $\delta_{i-1,j-1}$ is zero for all i,j except for the 1,1 entry, in which case $\delta_{0,0} = 1$.

During each control period, new values of α, g_1, g_2 are calculated, along with intermediate values, in the following order:

$$V_1 = R_{11}B_1 + R_{12}B_2 + Q_1 \tag{3.21}$$

$$V_2 = R_{12}^T B_1 + R_{22}B_2 + Q_2 \tag{3.22}$$

$$\beta = w_1 + w_2 + s + B_1^T(V_1 + Q_1) + B_2^T(V_2 + Q_2) \tag{3.23}$$

$$g_1 = A_1^T V_1/\beta, \quad g_2 = A_2^T V_2/\beta, \quad \alpha = w_2/\beta \tag{3.24}$$

$$R_{11}(k+1) = [\delta_{i-1,j-1}] + A_1^T R_{11}(k) A_1 - \beta g_1 g_1^T \tag{3.25}$$

$$R_{12}(k+1) = A_1^T R_{12}(k) A_2 \quad - \quad \beta g_1 g_2^T \tag{3.26}$$

$$R_{22}(k+1) = [\delta_{i-1,j-1}] + A_2^T R_{22}(k) A_2 - \beta g_2 g_2^T \tag{3.27}$$

$$s = (1-\alpha) w_2 \quad \text{and} \quad \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} = w_2 \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} \tag{3.28}$$

Both R_{11} and R_{22} must be maintained symmetric. This can be enforced by reassigning R_{11} as $0.5(R_{11}+R_{11}^T)$ and R_{22} as $0.5(R_{22}+R_{22}^T)$ after each iteration.

System to be Controlled

Mechanical System Example

Figure 3.2 shows a mass-spring system with control input u and disturbance input d. At designated points in time, spring constant k_3 can be changed from one selected value to another. Although not shown in the figure, viscous friction terms are included on each mass.

We use the following assignment of state variables:

$$x_1 = p_1, \quad x_2 = \dot{p}_1, \quad x_3 = p_2, \quad \text{and} \quad x_4 = \dot{p}_2$$

The resulting state variable description is

$$\dot{\mathbf{x}} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -(k_1+k_3) & -\beta_1 & k_1 & 0 \\ 0 & 0 & 0 & 1 \\ k_1 & 0 & -(k_1+k_2) & -\beta_2 \end{pmatrix} \mathbf{x} + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} u + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} d$$

where β_1 and β_2 are viscous damping coefficients. Feedback control is used to add damping to the system. Several different cases are considered. In one case, only a filtered version of $v_1 = \dot{p}_1$ is assumed to be available as the system output for use in identification and control; this corresponds to local feedback only. In another case, filtered versions of both $v_1 = \dot{p}_1$ and $v_2 = \dot{p}_2$ are available for identification and control, corresponding to regional control (Figure 3.3).

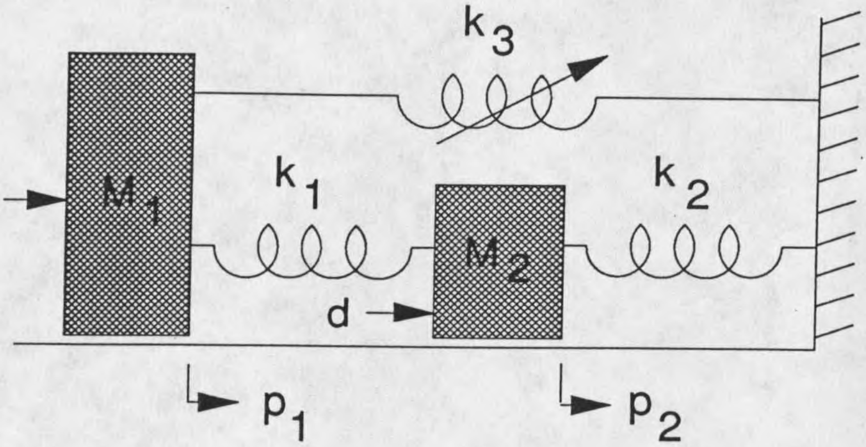

Figure 3.2. A mechanical system example.

The particular system parameters used in this study are M_1=M_2=1, β_1=β_2=0.1, k_1=k_2=10, and two values for k_3 which result in the following system transfer functions:

$$\frac{V_1}{U} = \frac{s(s+0.05+j4.472)(s+0.5-j4.472)}{D(s)}$$

and $V_2/U = 10s/D(s)$, where for k_3=15,

$$D(s) = [(s+0.05)^2+5.728^2][(s+0.05)^2+3.491^2]$$

and for k_3=0,

$$D(s) = [(s+0.05)^2+5.116^2][(s+0.05)^2+1.954^2]$$

For k_3=15, the system exhibits peaks in the frequency response at 0.9116 Hz and 0.5557 Hz. For k_3=0, these peaks are moved to 0.8143 Hz and 0.3110 Hz. The zeros of the system remain the same in either case. Note that the V_1/U zero has a frequency component of 0.7117 Hz which is somewhat centrally located between the poles when k_3=15.

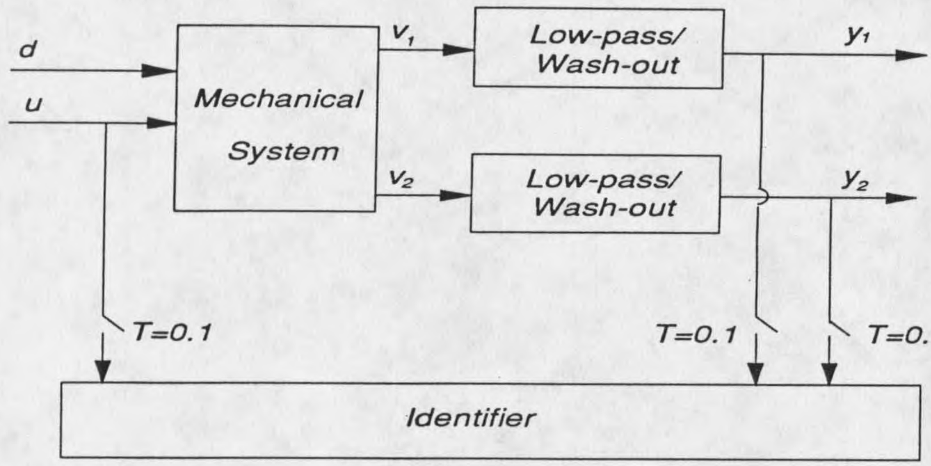

Figure 3.3. Total plant.

Output Filters

The first low-pass/wash-out filter of Figure 3.3 is characterized by two state variables, as follows:

$$\begin{pmatrix} \dot{x}_{1f} \\ \dot{x}_{2f} \end{pmatrix} = \begin{pmatrix} -31.4 & -31.4 \\ 0 & -0.25 \end{pmatrix} \begin{pmatrix} x_{1f} \\ x_{2f} \end{pmatrix} + \begin{pmatrix} 31.4 \\ 0.25 \end{pmatrix} v_1$$

with $y_1 = x_{1f}$. The second low-pass/wash-out filter is characterized in like fashion.

The noise input in Figure 3.2 and Figure 3.3 is white noise that is sampled and held (sample period T=0.1 s) with Gaussian distribution and variance that can be adjusted to different values to vary the signal-to-noise ratio. Additional details concerning the simulation of this system can be found in [57].

Test Results

Several different open-loop and closed-loop tests are described in this section. The probing signal used in the tests is a pseudo random rectangular wave, with zero average value and with an amplitude that is used as a test parameter. The probing signal is applied at the control input u of Figure 3.2. For the test results that involve noise inputs, the noise is applied at point d of Figure 3.2. For those tests in which no parameter changes are imposed on the plant, a value of 15 is used for k_3 (see Figure 3.2).

Open-Loop Tests

The output y_1 denotes the low-pass/wash-out filtered version of velocity v_1 of mass 1 in Figure 3.2. Figure 3.4 shows y_1 as a function of time when d=0 and the input $u(t)$ is a quasirandom probing signal which switches between +1.0 and -1.0 (magnitude=1.0). Figure 3.5 shows y_1 when u=0 and d is Gaussian white noise with variance of 1. Note that both outputs are comparable in amplitude, and that the response of the system to the noise is colored by the system.

In cases where $d \equiv 0$ and the input probing of magnitude 1 is used, RLS is very effective in obtaining accurate estimates of the transfer function coefficients. However, in Tables 3.1 and 3.2, RLS identified parameters are compared with the actual parameters under the following conditions: the probing magnitude is 1.0 and d is unity variance Gaussian white noise. For the results in Tables 3.1 through 3.4, the system output is the unfiltered version of the velocity v_1 and a $4th$-order model is assumed.

Results in Tables 3.3 and 3.4 were obtained under the same conditions as those used to obtain Tables 3.1 and 3.2, but using the RELS algorithm instead of the RLS algorithm.

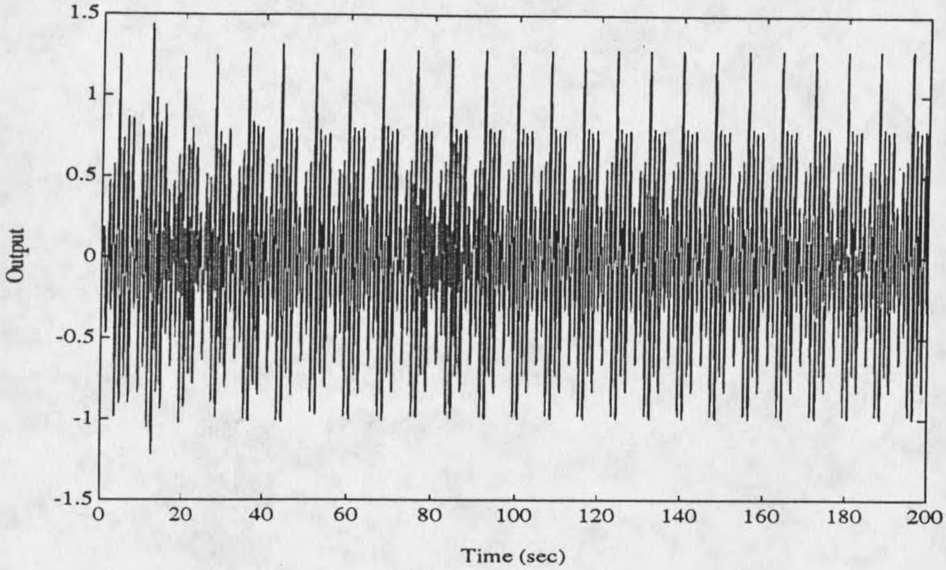

Figure 3.4. Output $y_1(t)$ (d=0 and probing magnitude=1).

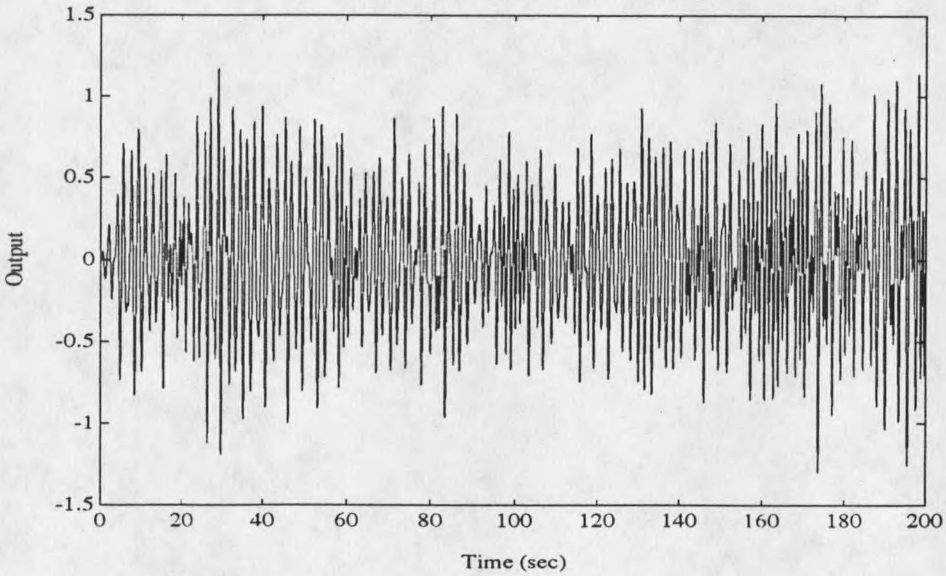

Figure 3.5. Output $y_1(t)$ (u=0 and white noise at d).

	a1	a2	a3	a4
System Parameters	-3.5424	5.1075	-3.5072	0.98020
Identified Parameters	-3.4145	4.7863	-3.2046	0.87769

Table 3.1. Denominator parameters (probing signal magnitude=1.0; noise variance=1.0; RLS).

	b1	b2	b3	b4
System Parameters	.095415	-0.26655	0.26560	-.094466
Identified Parameters	.095107	-0.25430	0.24373	-.084244

Table 3.2. Numerator parameters (probing signal magnitude=1.0; noise variance=1.0; RLS).

	a1	a2	a3	a4
System Parameters	-3.5424	5.1075	-3.5072	0.98020
Identified Parameters	-3.5464	5.1218	-3.5233	0.98736

Table 3.3. Denominator parameters (probing signal magnitude=1.0; noise variance=1.0; RELS).

	b1	b2	b3	b4
System Parameters	.095415	-0.26655	0.26560	-.094466
Identified Parameters	.095910	-0.26740	0.26676	-.095278

Table 3.4. Numerator parameters (probing signal magnitude=1.0; noise variance=1.0; RELS).

Closed-Loop Tests

In the following plots, system signals of interest are displayed under a variety of conditions imposed on the probing signal and on the noise input. In all cases, during an initial time interval of 20 seconds, *u(t)* is the pseudo random probing signal, identifiers and observers operate using sampled values of the probing input and system outputs, and the appropriate Riccati equations are iterated once per sample period. Closed-loop adaptive control starts at t=20. The initial values of model parameters are zero at t=0. Although 6th-order models are used in the results shown, 4th-order models give comparable results.

Closed-Loop Tests with System Parameters Constant

In cases where $d \equiv 0$, system oscillations are well damped by the adaptive control within a few seconds. Even with $d \equiv 0$, however, if the probing component of the control is discontinued while identification is continued, it is possible for the identified parameters to drift, and for bursting of the control to occur.

In Figure 3.6 the response corresponds to the following conditions: the probing signal (0.1 magnitude) is discontinued after t=20; unity variance Gaussian white noise is applied at d for all t; RLS identification is used; and adaptive control is active after t=20.

In Figure 3.7, the probing signal of 0.5 magnitude is used throughout the simulation; the probing action is added to the adaptive control action to form the $u(t)$ applied. In comparing results, continuous probing generally produces better results than those obtained with discontinuous probing. For each level of system noise a different level of probing signal is appropriate; this was verified for this system by viewing results obtained using several different probing and noise levels.

Closed-Loop Tests with Step Changes in Constant k_3

In the four closed-loop tests of this subsection, conditions that apply are: 1) a constant force of 2 units is superimposed on u for all t; 2) spring constant k_3 is alternated between 15 and 0, with switching times at t =100s, 400s, 800s, 1200s and 1600s; 3) the probing signal of magnitude 0.4 is continued during operation; and 4) d is unity variance Gaussian white noise.

Figure 3.8 was obtained using an RLS identifier, and the response illustrates temporary build up of instability at different points in time.

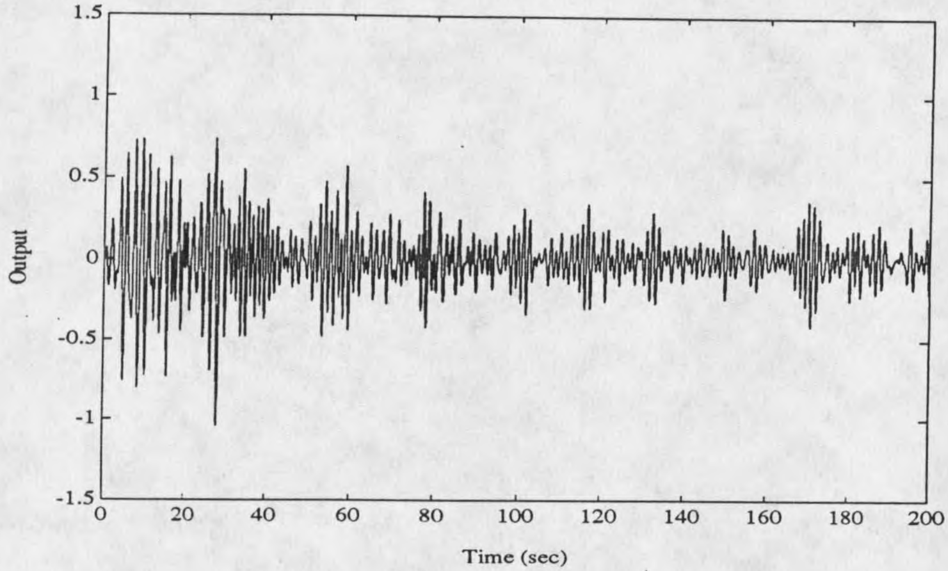

Figure 3.6. Output $y_1(t)$ (noise active; initial probing).

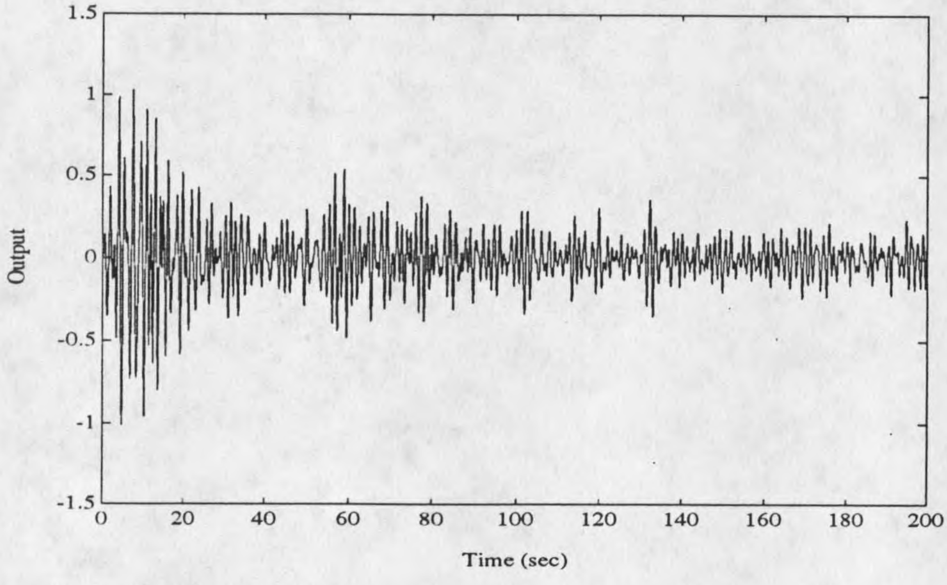

Figure 3.7. Output $y_1(t)$ (noise active; continued probing).

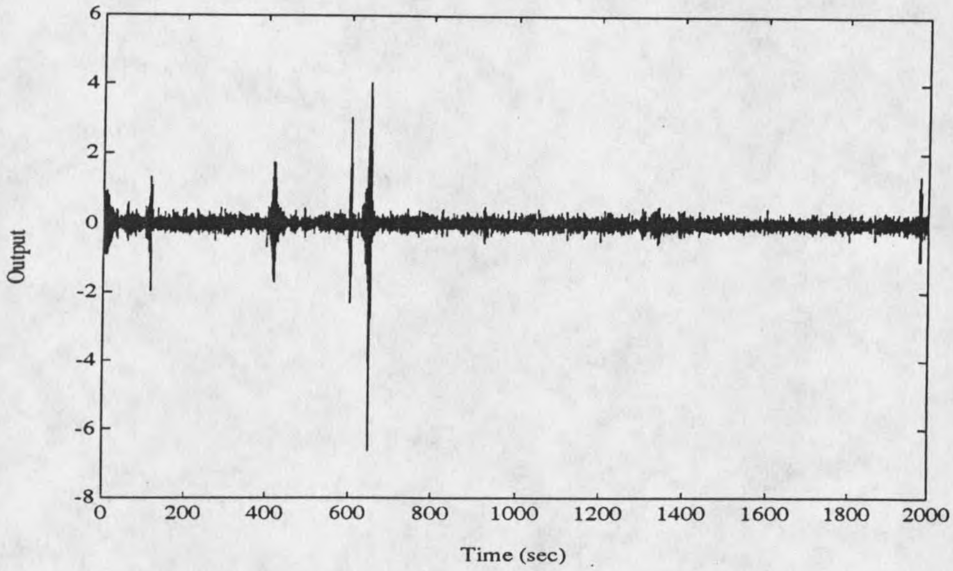

Figure 3.8. Output $y_1(t)$ (step changes in k_3; RLS identifier).

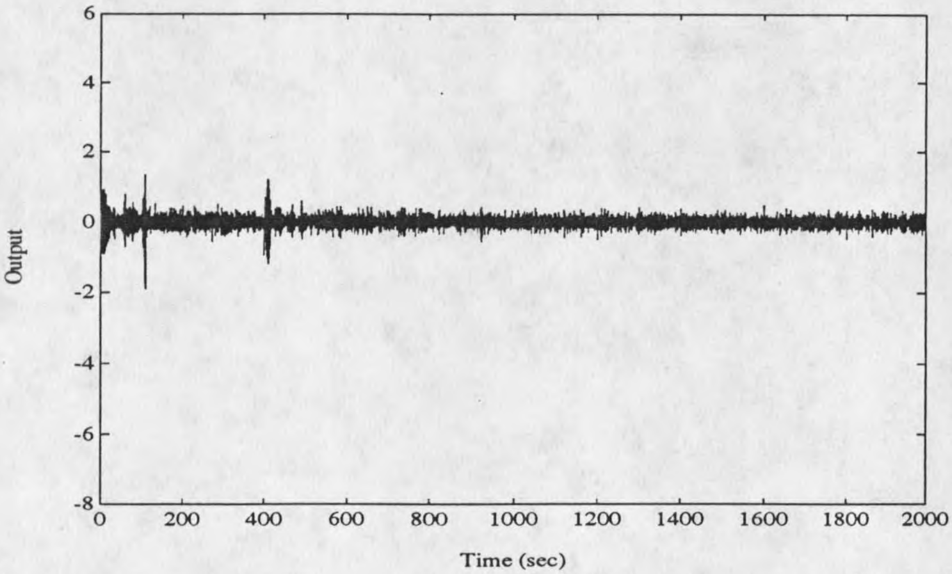

Figure 3.9. Output $y_1(t)$ (noise active; step changes in k_3;

RELS identifier; simplified observer).

Figure 3.9 was obtained using the RELS identifier, but with the C matrices and associated estimated error terms not included in the observer state equations.

Figure 3.10 was obtained using both the RELS identifier and the observer equations (3.10), (3.15), and (3.16). The slight deterioration in performance (Figure 3.10 versus Figure 3.9) was not anticipated.

Figure 3.11 was obtained using the estimated outputs of the identifiers, rather than the actual system outputs, for observer and control generation; other conditions were the same as those of the previous case. The estimated outputs provided sufficient filtering to eliminate the slight instability bursts.

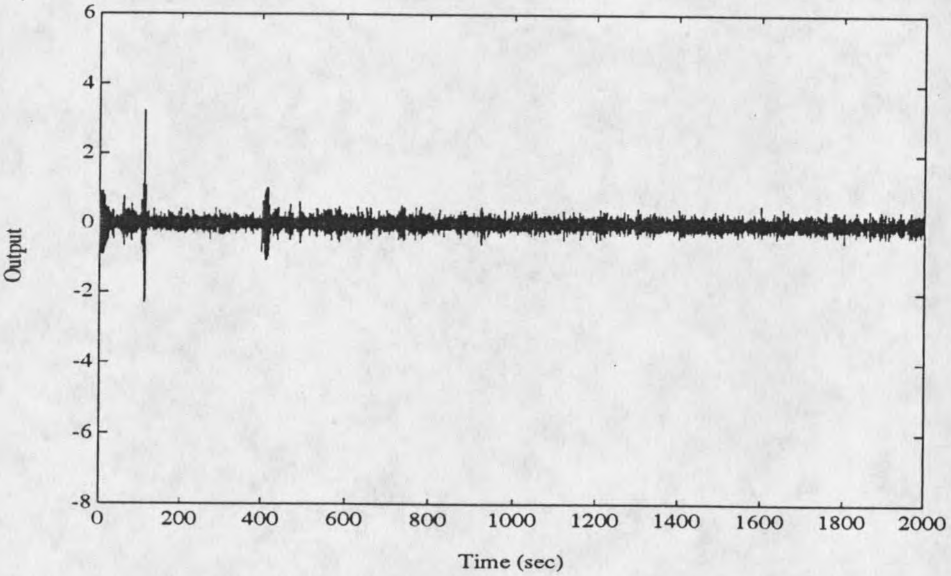

Figure 3.10. Output $y_1(t)$ (noise active; step changes in k_3; RELS identifier; complete observer).

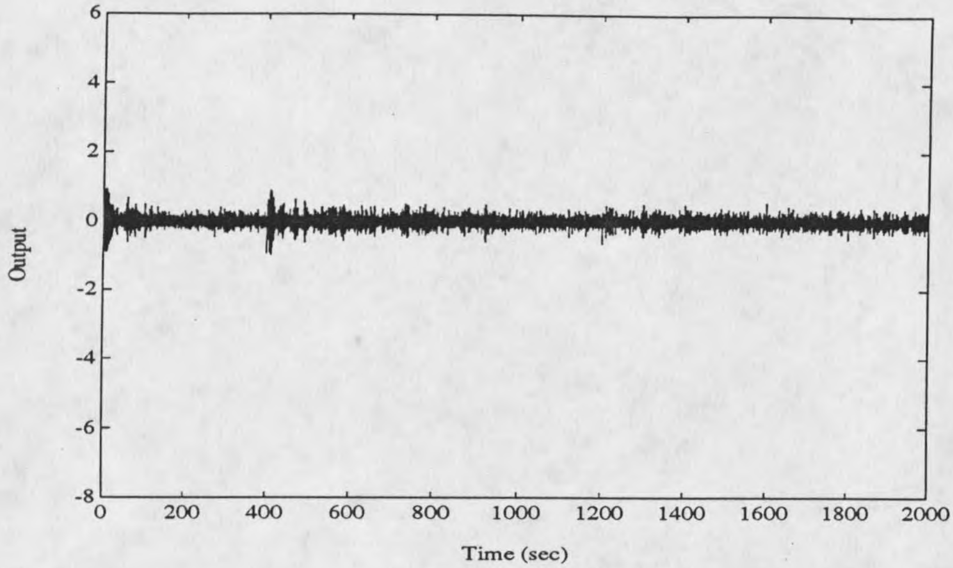

Figure 3.11. Output $y_1(t)$ (noise active; step change in k_3; RELS identifier; complete observer).

By increasing the level of probing from 0.4 magnitude to 0.6, the slight instability bursts of Figures 3.9 and 3.10 can be eliminated, and the responses corresponding to Figures 3.9, 3.10, and 3.11 can be made comparable. This is not the case, however, for the response in Figure 3.8 that corresponds to the use of the RLS identifier.

One-Output Feedback Versus Two-Output Feedback

In all the closed-loop tests conducted above, the adaptive controller makes use of both y_1 and y_2, low-pass wash-out filtered versions of velocities v_1 and v_2, respectively. Simulation results also were generated with the controller having access only to y_1 or to y_2. Generally more favorable results were obtained by using both system outputs in the control.

Discussion of Results

In order to assure robustness, most self-tuning adaptive controllers need to use continued probing to maintain a reasonable model of the plant. When there is no process noise in the system, this probing action may be observed in the system output and may be viewed as a disadvantage of such an adaptive controller. However, when there is process noise present, as is certainly the case in power systems, for example, the combined effect of continued probing and adaptive control is generally to decrease the oscillatory amplitudes in system signals. Of course, the key is to select a probing signal level that is neither too small (having insignificant impact compared to the noise) nor too large (driving system responses to amplitudes well beyond those produced by the noise alone).

In this work we have described a particular self-tuning adaptive controller and illustrated how it performs on a lightly damped mechanical system in the presence of a significant amount of process noise. Although the applied noise was white, the system itself colored the noise. A series of open-loop and closed-loop tests were conducted, and abrupt parameter changes were included.

The open-loop tests illustrated the response of the system to noise alone and to probing alone. The open-loop tests also illustrated properties of the identifier used in the adaptive controller: system parameters were estimated under conditions of no noise (probing only), and under conditions of approximately equal amounts of probing and noise. The RELS algorithm produced significantly better results than those obtained with the RLS algorithm.

A variety of closed-loop test results were presented. In cases where no noise was injected into the system, the adaptive controller performed admirably, damping the system oscillations within a few cycles. In cases where noise was injected, the amount of damping depended on the level of probing signal used. In those cases where noise was injected, the best results were obtained by superimposing probing of an appropriate level on the adaptive control, rather than discontinuing probing after the initial 20 seconds. For cases involving abrupt parameter changes, the use of continued probing was crucial to the elimination of instability bursts. The adaptive control remained robust when the assumed model order was reduced from 6th-order to 4th-order.

Many additional features are of interest regarding this adaptive controller. A supervisory structure should be imposed on this adaptive controller, and the resulting controller should be tested on simulations of noisy high-order systems, with identified model order necessarily much smaller than the system order. Another logical step for power system applications is to include suitable noise models in a standard power systems swing program, and to test this adaptive controller on a realistic power system simulation (see Chapter 4).

CHAPTER 4

EFFECTS OF NOISE ON THE ADAPTIVE DAMPING OF LOW-FREQUENCY OSCILLATIONS IN POWER SYSTEMS

In this chapter, the advanced adaptive controller described in Chapter 3 is applied to dampen low-frequency oscillations in a large interconnected power system simulation [24]. Robustness of the control is examined relative to realistic load noise and faults that change the system operating point. The controller uses information from two system outputs and controls a static var unit to dampen low-frequency oscillations. The identifier in the controller uses a version of RELS, and the identified parameters are used in an observer and in an enhanced linear quadratic (LQ) control law. Results are shown for a 2000-bus system that represents the western North American power system. The simulation contains realistic nonlinear machine models and additive load noise to represent actual operating conditions.

The organization of this chapter is as follows. First, the main features of the controller are summarized. Following this the characteristics of the test system are described. Test results and discussion are presented at the end.

Controller Structure

The adaptive controller consists of three major parts: 1) an RLS or an RELS identifier to identify coefficients of a difference equation model of the system being controlled; 2) an adaptive observer for obtaining state estimates of the system; and 3) an adaptive enhanced LQ controller. Figure 4.1 depicts the control system diagram for the case in which the system has two outputs y_1 and y_2, a control input u, and a disturbance input d that is not directly measurable. As indicated in the diagram, each identifier is supplied with sampled values of the input, u, and an output, y_1 or y_2. The identified parameters are supplied to a corresponding observer and to the Riccati gain generator, and the gain generator updates the controller gains for weighting the past controller output and the state estimates produced by the observers, resulting in the next controller output.

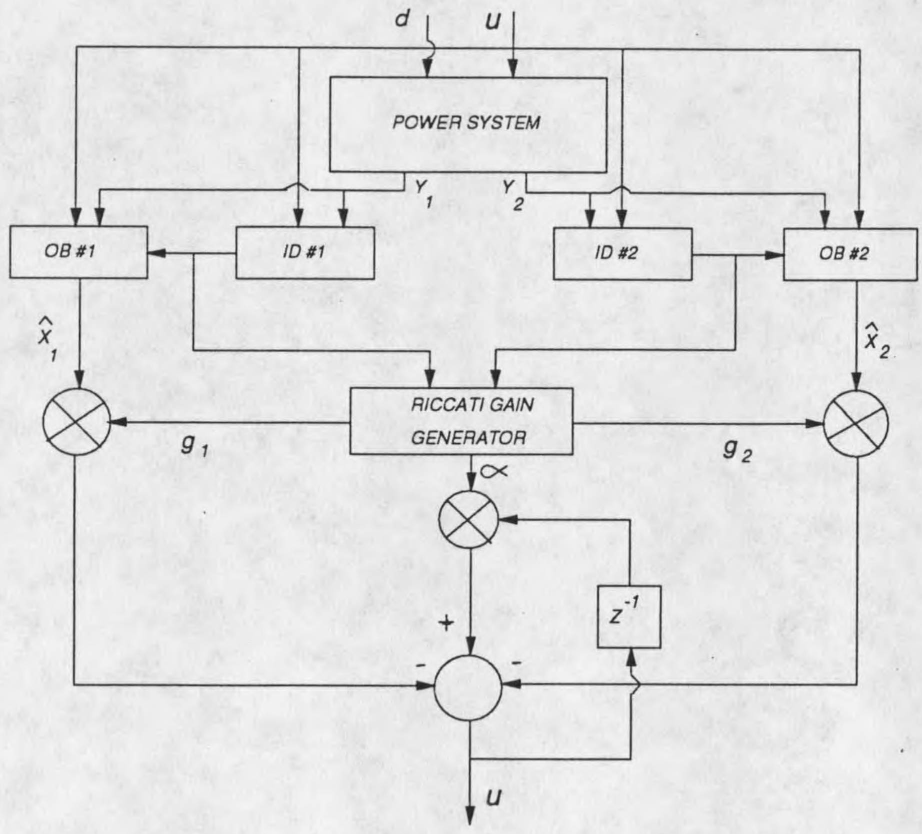

Figure 4.1. System diagram of an adaptive controller.

Test System Description

The power system example described here consists of an enhanced LQ adaptive controller applied to a static var compensator (SVC) in a large scale power system simulation. The simulated power system is a detailed production case used to study dynamic aspects of the entire western North American interconnected system. The case contains 2083 buses and 389 generators covering the region illustrated in Figure 4.2.

The adaptive controller is applied to an SVC located at a bus in the Seattle area. The SVC is a thyristor controlled device which can rapidly alter the effective shunt reactance seen by the network and thus control the voltage magnitude at the bus where it is located. The SVC in this case is rated to provide reactive power in the range from -250 MVAR to 250 MVAR. The main purpose of the SVC is local bus voltage control. The adaptive controller is added as a supplementary control loop to provide additional damping to interarea electromechanical oscillations. Two signals are used as inputs to the adaptive controller; one is the AC voltage at a bus located in the Seattle area and the other is the relative phase angle between the AC voltage at the CHIEF JO 230 bus and the AC voltage at a bus located in the Portland area. These signals generally exhibit one of the main modes involving the northern area of the network swinging against the southern area at low frequencies (0.3 Hz). The advantage of using the relative phase angle signal in this case is that it enhances the controllability of low frequency signals. The measurement of relative phase angles between different areas of large power systems is a subject of considerable recent activity among utilities. The Portland and Seattle areas are large urban areas of the Pacific northwest part of the network.

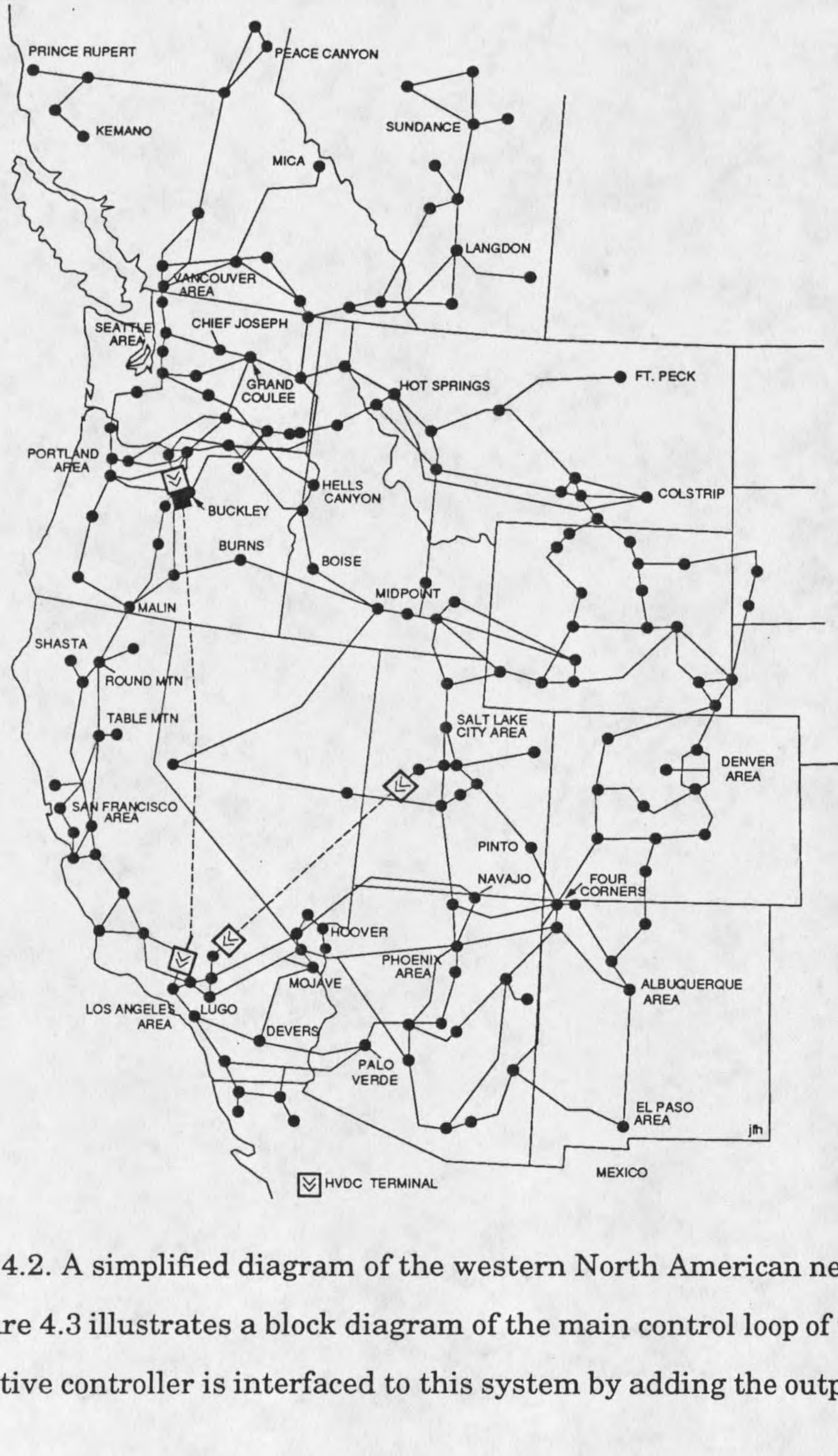

Figure 4.2. A simplified diagram of the western North American network.

Figure 4.3 illustrates a block diagram of the main control loop of the SVC. The adaptive controller is interfaced to this system by adding the output of the adaptive controller to the SVC voltage reference signal. This results in a "modulation" of the reference signal to produce additional damping of system oscillations. Fifth-order models were used in the adaptive controller.

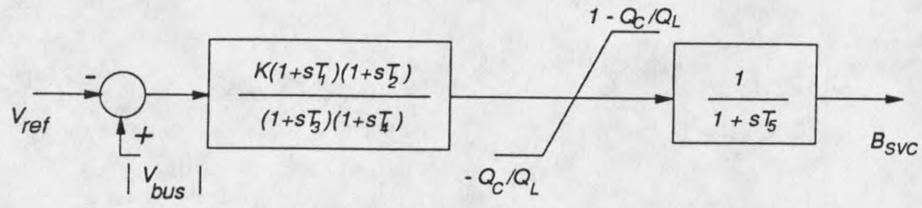

Figure 4.3. Block diagram of SVC main control loop.

Simulation Results

Several different tests are described in this section. For the test results that involve added system noise, an approximation of Gaussian white noise is applied to change the load at a major bus in the Seattle area; this white noise excites modes of the system and the excited response is viewed as colored noise. The applied noise approximates random load switching which invariably occurs in large power systems. Figure 4.4 shows the time response of the applied noise, and Figure 4.5 displays its associated FFT.

The effect of this applied noise on signals in the system is of interest. For example, Figure 4.6 displays the CHIEF JO relative phase angle signal, both with and without the applied load noise, under the following conditions: a three-phase fault is applied at the MALIN 500 bus which is located on a major AC intertie that connects the Pacific northwest to California; the fault is initiated 6 seconds after the start of the simulation and lasts for 0.15 seconds; and the adaptive control is inactive. Figure 4.7 displays FFT's of the signals in Figure 4.6.

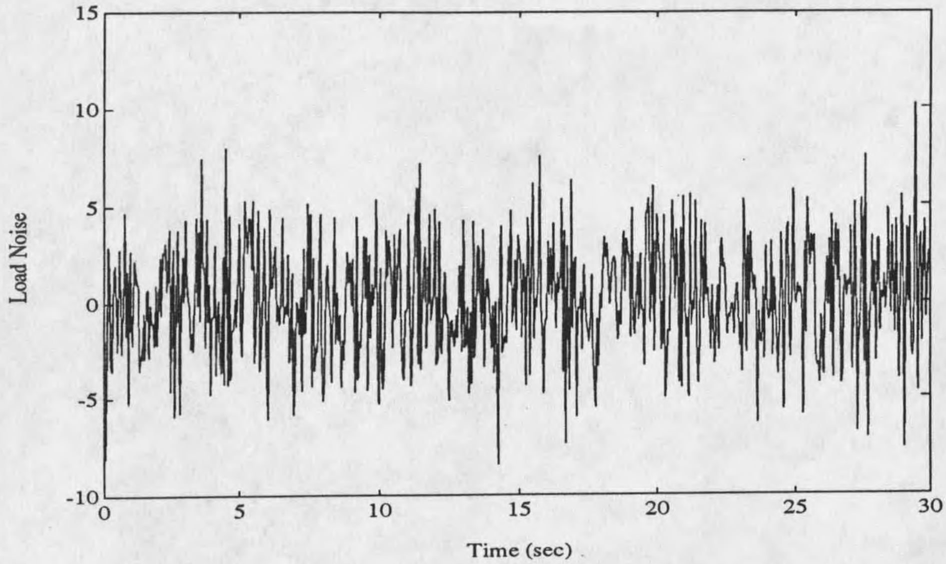

Figure 4.4. Time response of the noise applied to the system.

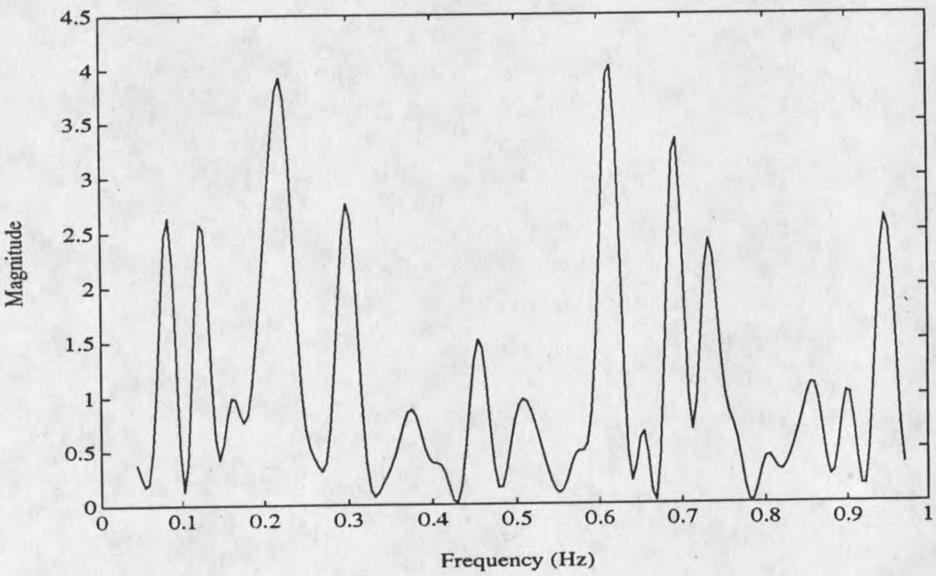

Figure 4.5. FFT of the noise applied to the system.

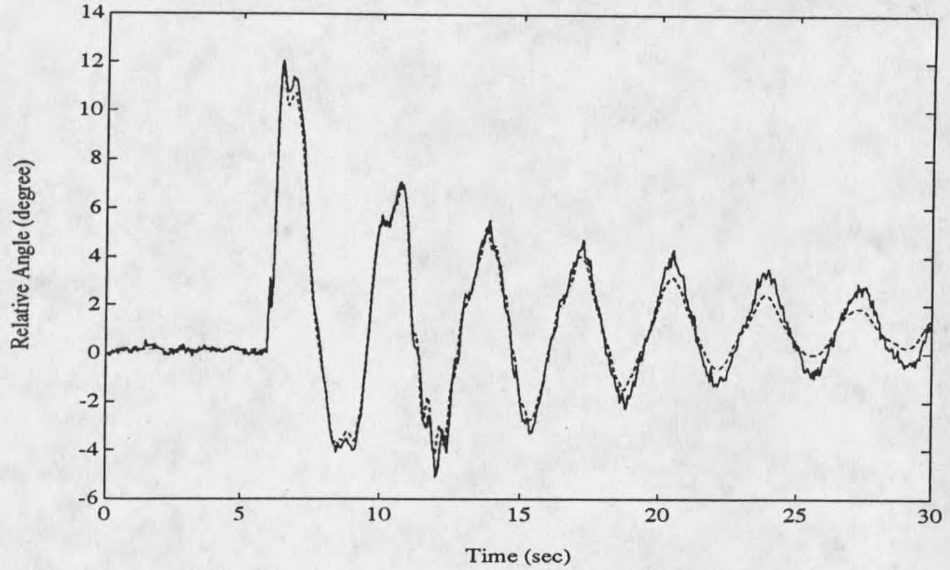

Figure 4.6. Relative phase angle signal with noise (solid) and without noise (dotted) without control.

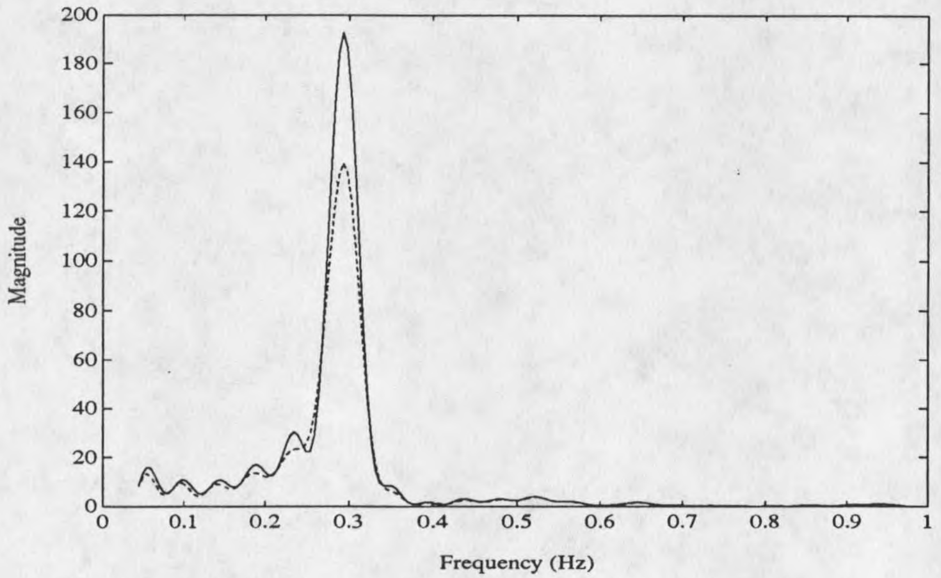

Figure 4.7. FFT of relative phase angle observed at bus CHIEF JO with noise (solid) and without noise (dotted) without control.

Control without Noise

Under the fault conditions described in the preceding paragraph, Figure 4.8 illustrates the effectiveness of the adaptive controller in damping oscillations that are observed in the relative phase angle signal between CHIEF JO and the Portland area. Figure 4.9 shows the effect of using an RLS identifier. In comparing results of using an RELS identifier with those obtained using an RLS identifier, the responses are similar (compare Figure 4.8 with Figure 4.9) when supplemental noise is not included in the simulation. Figure 4.10 illustrates the effectiveness of the same adaptive controller at damping out oscillations which are observed in the relative phase angle in the Arizona area due to the disturbance at MALIN 500. Thus, a 0.3 Hz mode between the north and the south of the power system can be controlled by this adaptive controller. Figure 4.11 compares the voltage at the bus CHIEF JO with (solid) and without (dotted) control.

Control with Noise

Figures 4.12 and 4.13 were obtained under the following conditions: 1) white noise with Gaussian distribution was sampled and held (sample period T=0.1 s) and added to the load to simulate random load fluctuations, and 2) for Figure 4.12 the RELS identifier and for Figure 4.13 the RLS identifier were used. Figure 4.13 illustrates poor performance of the adaptive controller (with the RLS identifier) because of noise.

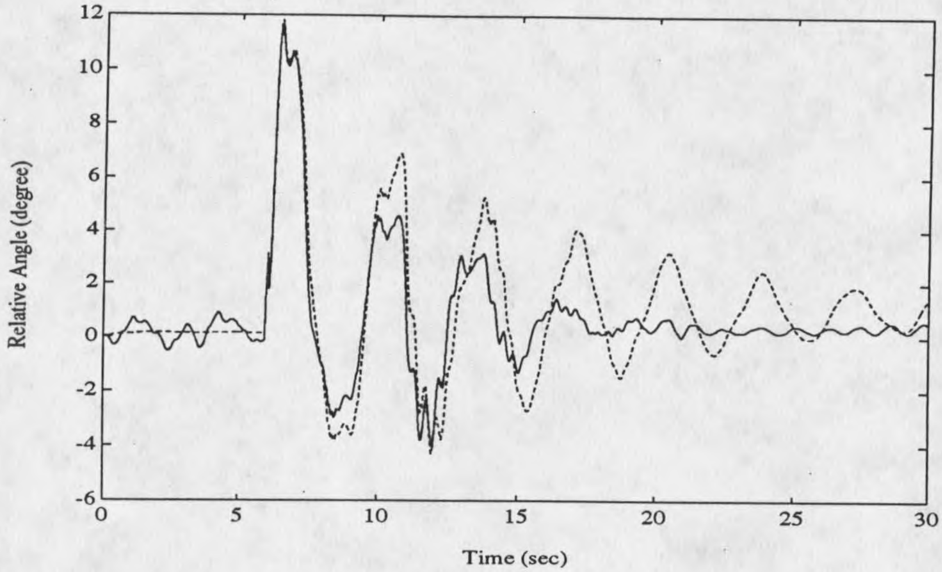

Figure 4.8. Relative phase angle signal with (solid) and without (dotted) control (RELS identifier).

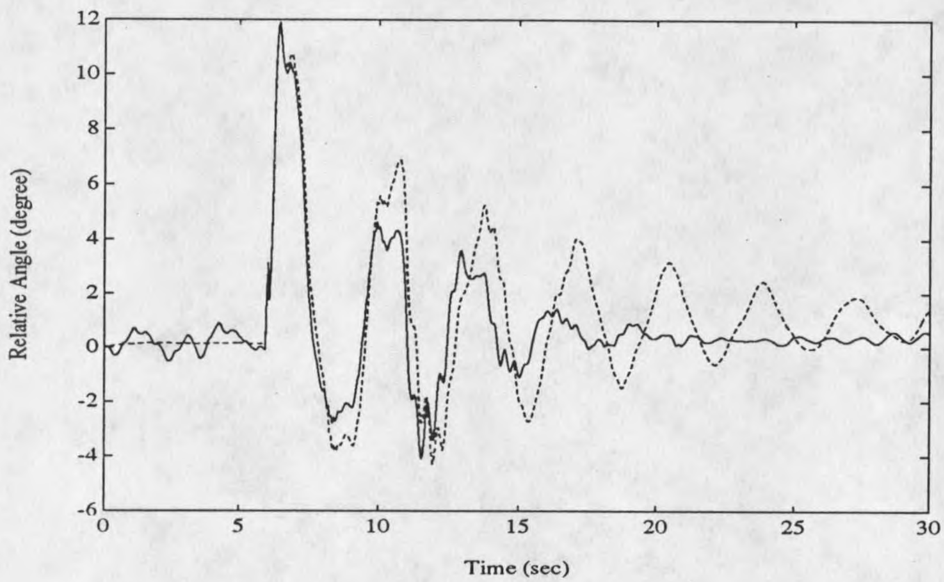

Figure 4.9. Relative phase angle signal with (solid) and without (dotted) control (RLS identifier).

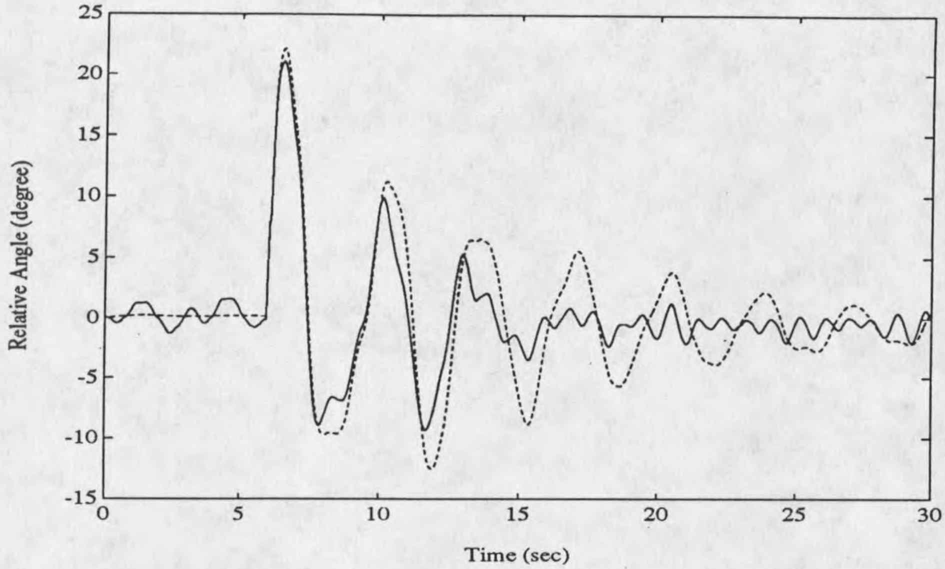

Figure 4.10. Relative phase angle signal with (solid) and without (dotted) control (RELS identifier).

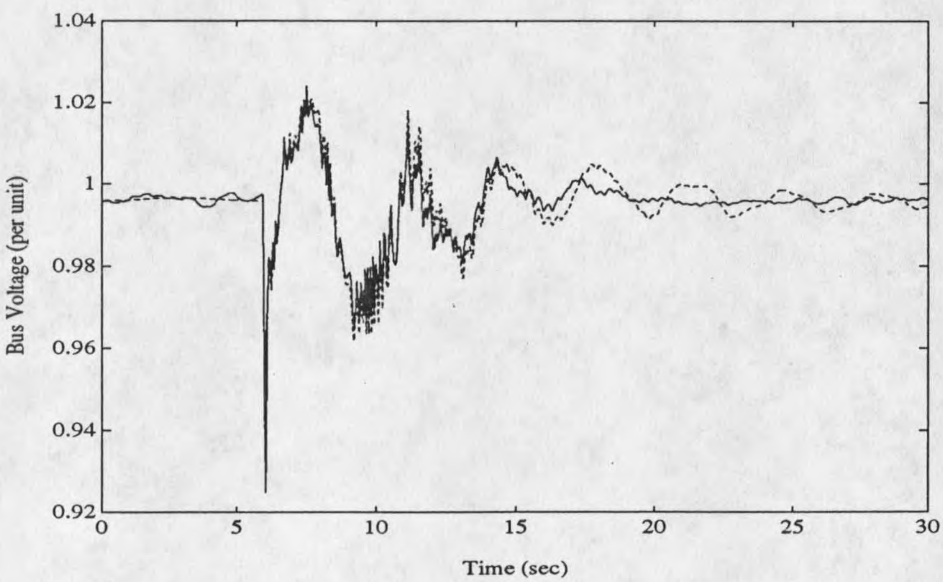

Figure 4.11. Voltage signal with (solid) and without (dotted) control (RELS identifier).

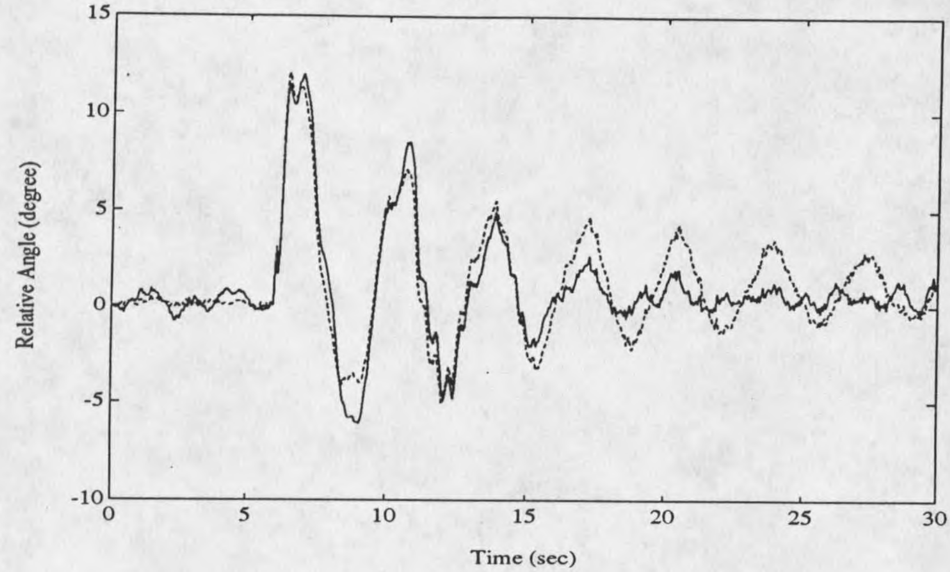

Figure 4.12. Relative phase angle signal with (solid) and without (dotted) control (RELS identifier).

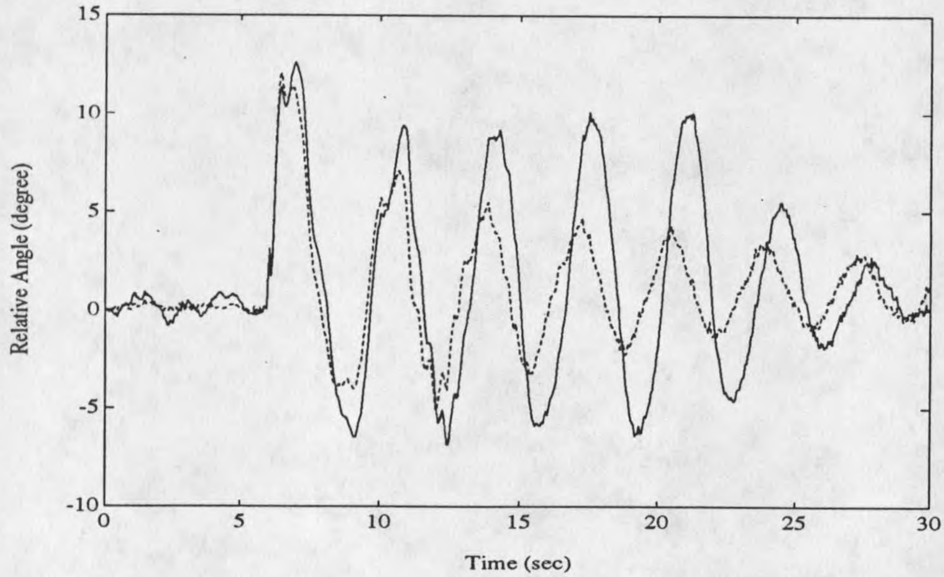

Figure 4.13. Relative phase angle signal with (solid) and without (dotted) control (RLS identifier).

Control with Noise and Continued Probing

In Figure 4.14 the probing signal was continued along with the adaptive control during the simulation (solid) and is compared with results from discontinued probing (dotted); other conditions were the same as in the previous case. In comparing results of continued probing with those of discontinued probing, very little difference is observed in this case. Other examples, however, have illustrated advantages for continued probing.

Control without Noise and Different Operating Point

In Figure 4.15, the operating point of the power system was changed by removing a line between GRIZZLY 500 and MALIN 500 which connects the Pacific northwest with California. The solid line illustrates the beneficial effect of the adaptive control compared to the no control case (dotted).

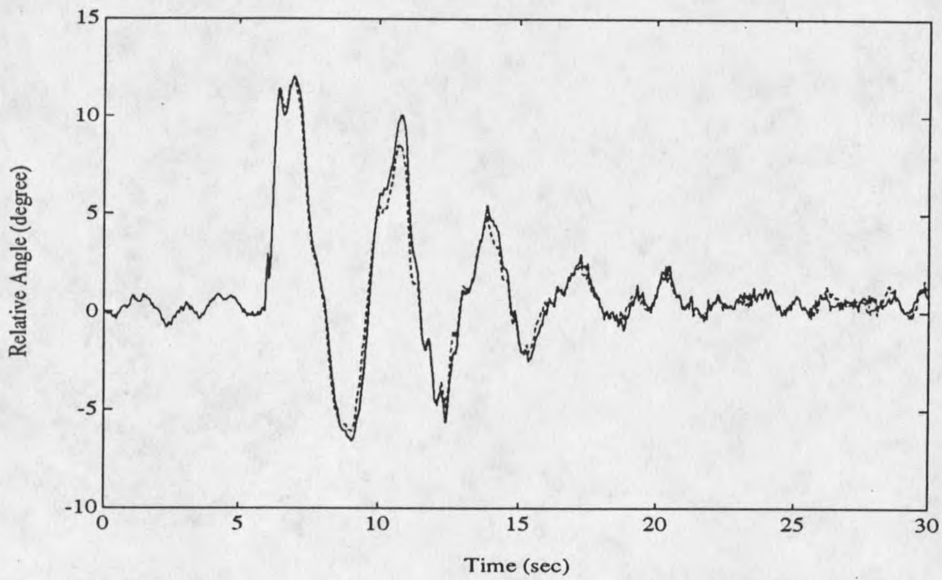

Figure 4.14. Relative phase angle signal with continued probing (solid) and with discontinued probing (dotted).

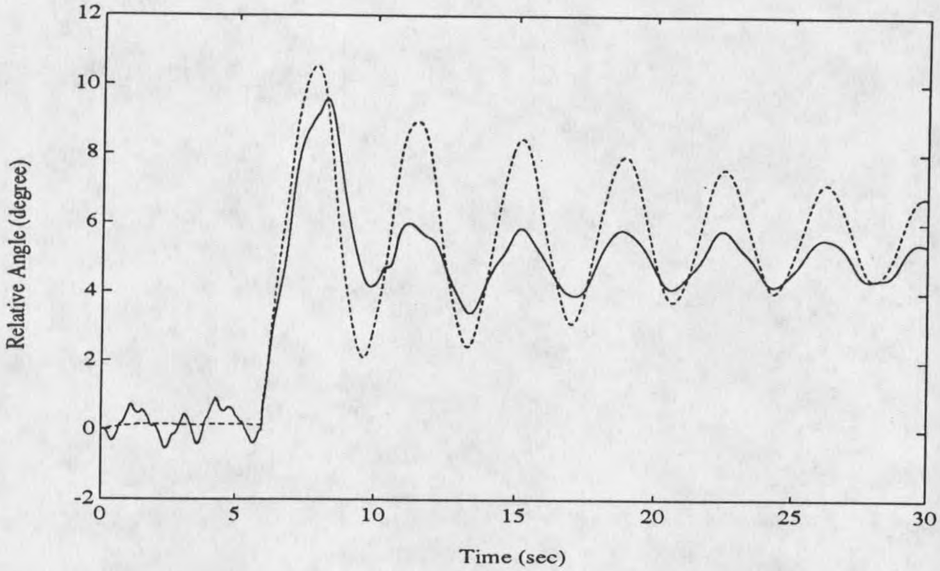

Figure 4.15. Relative phase angle signal with (solid) and without (dotted) control (different operating point).

Control with Noise and Different Operating Point

Figures 4.16 and 4.17 were obtained under the following conditions: 1) white noise with Gaussian distribution was sampled and held (sample period T=0.1 s) and added to the load to simulate random load fluctuations, and 2) the operating point of the power system was changed by removing a line between GRIZZLY 500 and MALIN 500 which connects the Pacific northwest to the California area; and 3) for Figure 4.16 the RELS identifier and for Figure 4.17 the RLS identifier were used. Better response is obtained when the RELS identifier is used (compare Figure 4.16 with Figure 4.17).

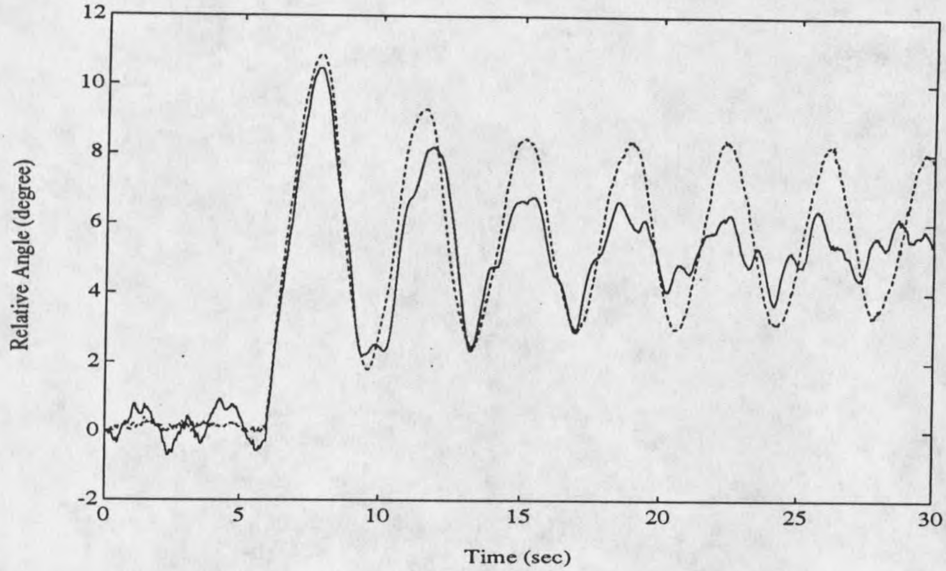

Figure 4.16. Relative phase angle signal with (solid) and without (dotted) control (RELS identifier).

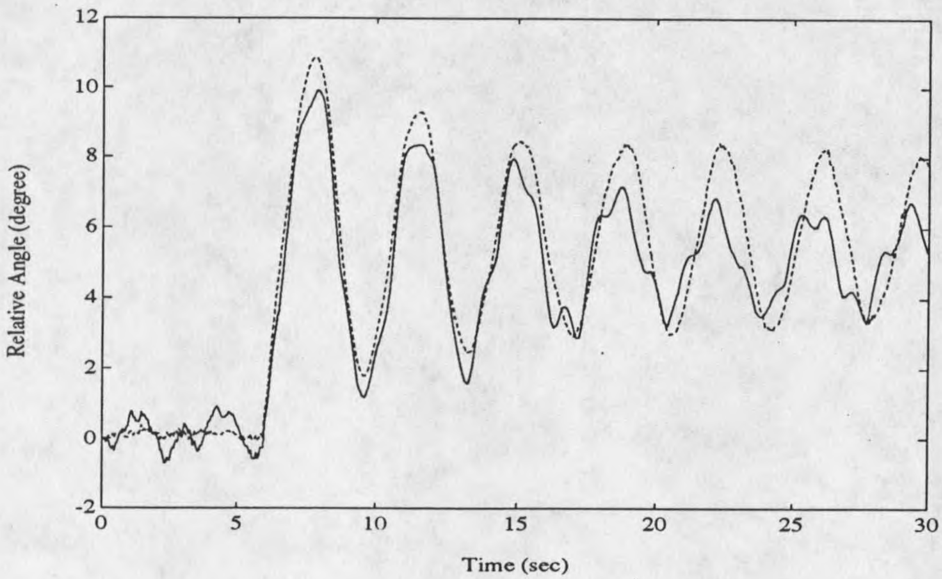

Figure 4.17. Relative phase angle signal with (solid) and without (dotted) control (RLS identifier).

Discussion of Results

In this chapter we have applied the self-tuning adaptive controller of Chapter 3 to dampen low-frequency oscillations in simulations of large-scale power systems. Cases with and without noise included were used to illustrate the robustness of the controller to system noise; in cases with substantial system noise, the RELS identifier generally produced better results than those obtained with the RLS identifier.

In the presence of process noise, self-tuning adaptive controllers needed to use continued probing to maintain a reasonable model of the plant. The combined effect of the process noise, continued probing, and adaptive control generally decreased the oscillatory amplitudes in system signals. The key was to select a probing signal level that was neither too small (having insignificant impact compared to the noise) nor too large (driving system responses to amplitudes well beyond those produced by the noise alone).

While the results displayed in this chapter are encouraging, there are uncertainties associated with the application of adaptive control in large power systems. Both the power system and the control exhibit many characteristics. For the control, adjustments can be made in many features, including performance weights, identifiers properties, probing level, SVC parameters, and the types of signals used for feedback. The power system structure and noise level can vary dramatically. Some changes can cause the adaptively controlled SVC to enter a limit cycle producing large oscillations throughout the power system. A significant amount of trial and error is needed to determine appropriate parameters for successful operation of the adaptive controller in this environment. Also, in some instances, disturbances can result in oscillations which may not be controllable by a particular SVC. In the cases illustrated in this chapter, however, the controller was effective in damping the dominant 0.3 Hz mode.

CHAPTER 5

TRANSFER FUNCTION IDENTIFICATION IN POWER SYSTEM APPLICATIONS

The purpose of this chapter is to present some methods and techniques of least-squares transfer function identification (TFI) in power system applications [29]. The primary application areas to be discussed include SVC tuning for interarea damping control, model validation and software validation. Several aspects dealing with the accuracy and validity of numerically identified transfer functions will also be discussed.

The main emphasis of this chapter is on identification of transfer functions in power systems using deterministic input signals and recently developed signal processing techniques. Transfer function identification problems in power systems are generally characterized by several factors. For example, the effective order of the system is unknown and can be large; the system has many nonlinearities which one has to be cautious about in applications; and the signals being used may contain colored noise which can bias the identified transfer functions.

The order of any transfer function in a large power system could be as large as the number of states that are needed to model the system. For most large systems this would mean many thousands of states. However, most of the system poles or eigenvalues cannot be observed from any one signal in the system. This situation allows one to identify the effective transfer function between any two points in the system. An input signal applied to a power system will usually excite many modes which are then exhibited to differing degree in different output signals and locations in the system. The effective transfer function which is identified by the techniques proposed here should contain enough of the pertinent dynamics to accurately model the effects of feedback within a linear range of operation.

The transfer functions being discussed here can be either Laplace transform (s-plane) transfer functions or z-transform (z-plane) transfer functions, depending on assumptions regarding the nature of the input signal. The transfer function between two points contains the poles (or eigenvalues) and zeroes which are acted upon by the input signal to produce an output signal. A transfer function is usually written as a ratio of polynomials but can also be expressed in parallel form as a sum of residues over first-order poles. The conversion from the rational polynomial form to the parallel form is the well known process of partial fraction expansion [58]. How well an identified transfer function reproduces output signals for specific inputs is a measure of the accuracy of the transfer function in modeling the observed phenomena. Another test of the accuracy of the transfer function is in how well it predicts the shift in pole locations that occur when a feedback loop is closed. Both of these tests are important for validating transfer functions which are obtained from measured signals by numerical methods.

Important applications of TFI methods in power systems include the following:

1. Tuning, design and testing of control systems. These include power system stabilizer (PSS) units, static var compensators (SVC), high voltage DC (HVDC) systems, subsynchronous resonance (SSR) devices and Flexible AC Transmission system (FACTS) devices.

2. Control signal analysis applications. This application involves the study of different controller input signals to determine which signal will provide the best results as a feedback signal for a particular system. This can be done by estimating the open-loop transfer functions for different signals and using a root locus analysis to estimate the effects of feedback.

3. Software validation. Utilities are increasingly faced with an emphasis on accurate system simulations in order to make short term and long term planning and operating decisions. TFI methods are very effective ways to make independent verification of any simulations of linear models contained within the power system network. Examples of these include the linear control systems used on governors, exciters, PSS units, SVC's, or HVDC systems. TFI methods can identify transfer functions from signals obtained from simulations and thus provide verification of the correctness of the software being used to carry out simulations.

4. Model validation. Equally important to software validation is model validation. The term 'model' is adopted here to refer to the data which is used to describe the dynamics of some part of the system. Since the data used in a model will determine the transfer function from a reference input to an output of the system, this can be verified by running a TFI study on the actual system to verify that the model data being used in simulations is accurate. In this application the identified transfer function obtained from signals measured on an actual control system can be compared to data being used for simulations. The same methods can be used to periodically check the control setting of a critical device. This information can assure that expensive equipment is being used to obtain maximum results.

5. Robustness and uncertainty measurements. Another application of TFI methods is to obtain a set of transfer functions for a range of operating conditions. This set of transfer functions will then give some information about the variability of the control environment that a particular control system will be required to operate under. This type of information can be used to incorporate robust control design techniques when such a consideration is warranted.

This chapter will briefly cover two basic identification methods, some application examples, and a discussion of validation tests which can apply to any technique.

Signal Identification and Small Signal Analysis

The purpose of this section is to discuss the similarities and differences among transfer function identification (TFI) methods, signal identification methods, and small signal analysis methods, for obtaining transfer functions and system eigenvalues. All three of these methods; TFI, signal identification, and small signal analysis, share a considerable overlap in information obtained from the results but each method occupies a slightly different niche in applications.

In addition to the least-squares identification methods of obtaining transfer functions, another approach often used is linearization of the system state equations around an operating point. These state equation methods, often referred to as small signal methods, are presented in [52,59]. The TFI methods and small signal methods can essentially provide the same information, but they differ in the data they use. TFI methods can be used on actual measured data as well as on simulated signals derived from simulation programs such as stability programs and electromagnetic transients programs. Transfer functions obtained from small signal analysis, as in [60], depend on stability data used to arrive at results. A significant advantage of the small signal approach is the type of global information which can be obtained about power system dynamics. By contrast the TFI methods provide more local information in the sense that the transfer function can give information about how one particular input affects an interarea mode but it does not give any information about what other factors may affect this mode. The primary niche filled by TFI methods is the ability to operate on actual measured system data without the need for any modeling data. This situation makes the TFI methods very useful in validating simulation software and computer models using signals obtained from field recordings.

TFI methods require knowledge of the input signal in order to compute the transfer function numerator polynomial. In many power system situations the input signal may be unknown or difficult to describe mathematically. In these cases the eigenvalues of the system can still be found by extracting them from system signals using signal identification methods such as those described in [48, 61-62]. In some signal processing application such as radar [63], these are also referred to as signal decomposition methods. Signal identification methods are often very similar to TFI methods in that they use the same numerical techniques such as singular value decomposition. The main difference is that signal identification methods do not use information about the system input in obtaining their results. Using these methods to analyze signals gives eigenvalues contained in the signal. If no input signal is applied during the time series being analyzed, then the eigenvalues contained in the signal are the same eigenvalues as contained in the system transfer function. Signal identification methods do not however directly give transfer function residues or zeroes. Signal identification methods can be applied when unexpected disturbances excite many system modes. In these situations the input signal may be unknown or difficult to describe accurately in a mathematical form. The use of signal identification methods on system signals which are active as a result of a fault can yield important information about the system frequencies and damping. Thus signal identification methods are important tools for real-time monitoring of system performance.

The main point to be made here is that all three methods (TFI, signal identification, and small signal analysis) provide similar types of information about a power system, but under different application circumstances. It is the author's experience that a detailed study of power system dynamics will often benefit from the utilization of all three of these analytical tools.

Transfer Function Identification Methods

A brief overview of two methods for transfer function identification will be presented. Both of the methods are based on the least-squares solution to an over-determined set of equations. The first method is for directly identifying what is often referred to as an auto-regressive moving average (ARMA) model. This will be referred to as a direct ARMA TFI method. The second method is a version of an extension of the Prony signal identification method to allow for incorporating input signal information to determine the transfer function parameters. This method will be referred to as the Prony TFI method. Both of these methods have many extensions and refinements but only the basic ideas will be presented here.

Direct ARMA

The direct ARMA TFI method starts with a z-transform description of the system transfer function $G(z)$ between an input signal $u(k)$ and an output signal $y(k)$:

$$G(z) = \frac{Y(z)}{U(z)} = \frac{b_1 z^{-1} + b_2 z^{-2} + \ldots + b_n z^{-n}}{1 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_n z^{-n}} \tag{5.1}$$

or, transfer function corresponds to the difference equation

$$\begin{aligned} y(k) = &- a_1 y(k-1) - a_2 y(k-2) - \ldots - a_{n-1} y(k-n+1) - a_n y(k-n) \\ &+ b_1 u(k-1) + b_2 u(k-2) + \ldots + b_{n-1} u(k-n+1) + b_n u(k-n) \end{aligned} \tag{5.2}$$

If there are N consecutive data samples, then N-n equations like (5.2) can be written, where n is the order of the transfer function. These N-n equations can be put in matrix form to solve for the unknown coefficients. The matrix equations have the form

$$Y = DP \tag{5.3}$$

where

$$Y = \begin{pmatrix} y(k) \\ y(k+1) \\ y(k+2) \\ \cdot \\ \cdot \\ \cdot \\ y(k+N) \end{pmatrix} \qquad \text{and} \qquad P = \begin{pmatrix} -a_1 \\ -a_2 \\ -a_3 \\ \cdot \\ \cdot \\ \cdot \\ -a_n \\ b_1 \\ b_2 \\ \cdot \\ \cdot \\ \cdot \\ b_n \end{pmatrix}$$

and the D matrix has the form

$$\begin{pmatrix} y(k-1) & y(k-2) & \dots & y(k-n) & u(k-1) & u(k-2) & \dots & u(k-n) \\ y(k) & y(k-1) & \dots & y(k-n+1) & u(k) & u(k-1) & \dots & u(k-n+1) \\ \dots & & & & & & & \\ \dots & & & & & & & \\ \dots & & & & & & & \\ y(k+N-1) & y(k+N-2) & \dots & y(k+N-n) & u(k+N-1) & u(k+N-2) & \dots & u(k+N-n) \end{pmatrix}$$

As was mentioned earlier, the order of the effective transfer function in power systems is usually not known. There is an interesting empirical relation, [64] which states that the best results are obtained when the D matrix above has about two to three times more rows than columns. Our experience in using this technique has supported this rule of thumb. The implication here is that the initial order of the transfer function used is based entirely on the number of data points taken. For this case then we would choose n to be an integer value of approximately N/6. The above over-determined set of equations can be solved by a variety of numerical techniques utilizing the pseudo-inverse of D, [65], and singular value decomposition [66]. These numerical algorithms are widely available in numerical software packages such as Linpack or MATLAB.

The transfer function as obtained at this point is usually of much higher order than is necessary and will likely contain numerous extraneous eigenvalues. If the fit is good the extraneous eigenvalues will be canceled by zeroes in the transfer function numerator. One method of reducing the order of the fitted model is to root both the numerator and denominator polynomials and cancel poles which are very near to zeros. Many extraneous poles can also be recognized by the fact they often appear at high frequencies which are known to be absent in the recorded signals.

Once a z-plane transfer function is obtained, it can be used as is or transformed into an s-plane transfer function. There are many methods for doing this transformation which differ in the assumptions and uses of hold circuits in the continuous time model.

Prony TFI

The second method to be discussed is the Prony TFI which is an extension of the Prony signal identification method presented in [48]. This method is very similar to the direct ARMA method in the way it uses the same numerical procedures to solve for an over-determined set of equations. The main difference is in how the equations are set up. Extensive research has been conducted on numerical techniques for solving the Prony analysis problem. An up-to-date description of many of these methods is contained in [67]. Significant research is continuing in this area with the most recent advances being the estimation of error bounds on identified parameters [68].

In the Prony TFI method the equations are set up in pole residue form. This is also sometimes referred to as parallel form. Given either an s-plane or a z-plane signal description, expressed as a ratio of polynomials where the denominator is of higher order than the numerator, the parallel form is obtained by partial fraction expansion. This can be expressed as

$$\hat{G}(s) = \sum_{i=1}^{n} \frac{\hat{R}_i}{s - p_i} \tag{5.4}$$

A corresponding zero-order-hold z-plane representation (see Figure 5.1) is $G(z)$:

$$G(z) = \sum_{i=1}^{n} \frac{R_i}{z - \lambda_i} \tag{5.5}$$

where

$$R_i = \frac{\hat{R}_i}{p_i}(\lambda_i - 1) \quad , \quad \lambda_i = e^{p_i T}$$

and T is a constant sampling period. $\hat{G}(s)$ and $G(z)$ are system transfer functions, the p_i's are s-plane eigenvalues, the λ_i's are z-plane eigenvalues, and $\hat{R}_i$'s and R_i's are transfer function residues.

Suppose in (5.2) that the input is applied at time k=1 and the time k = d+1 is the first sample value for which the input is no longer being applied. When no input is being applied, then the signal 'ring down' can be described by a difference equation model where the output is dependent only on past outputs and not on any inputs. This type of model is commonly referred to as an auto regressive model or AR model. In this case (5.2) becomes a homogeneous difference equation of the form

$$y(k) = -a_1 y(k-1) - a_2 y(k-2) - \dots \\ -a_{n-1} y(k-n+1) - a_n y(k-n) \qquad for \quad k > n+d \tag{5.6}$$

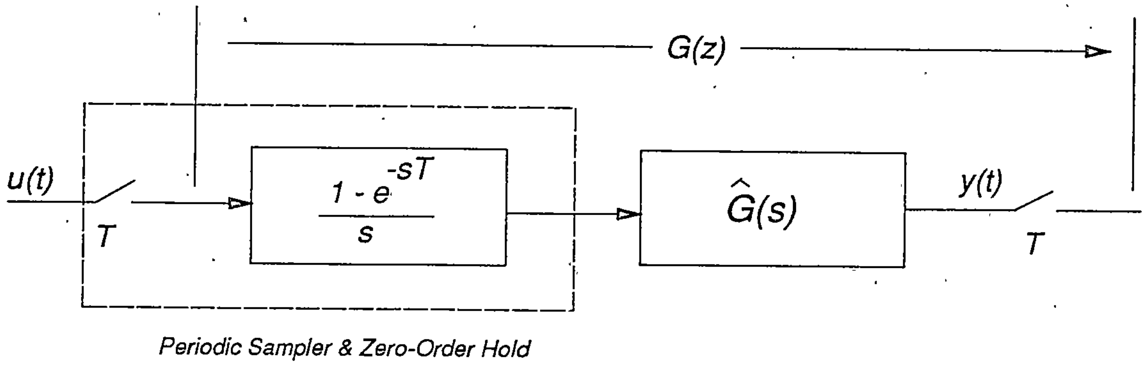

Figure 5.1. Continuous system transfer function $\hat{G}(s)$ and a corresponding discrete-time transfer function $G(z)$

The Prony TFI method uses a matrix form very similar to that in (5.3) to solve for the coefficients of this difference equation. Once these coefficients are obtained the characteristic polynomial associated with (5.6) is rooted to obtain the transfer function eigenvalues, the λ_j's. The solution of the homogeneous difference equation (5.6) is then given by

$$y(k) = \sum_{j=1}^{n} \overline{R}_j \lambda_j^k \qquad for \quad k > d \tag{5.7}$$

where the λ_j's are the eigenvalues of the discrete transfer function G(z) and the $\overline{R}_i$'s are the signal residues of the ring down portion of the output signal. These residues are a function of the input signal as will be discussed shortly. The residues are obtained by using (5.7) in the right-hand-side of (5.6) to set up a second set of over determined equations in the form of (5.3), but with the vector P containing the unknown $\overline{R}_j$'s in this case, and again using the pseudo-inverse to solve for the unknown parameters [65].

At this point knowledge of the input signal can be utilized to calculate transfer function residues from the signal residues which have been identified. In order to find the system transfer function we need to know the poles and zeros of the system, or the poles and residues associated with the impulse response of the system. If the input signal is restricted to be block pulses whose heights and duration are known then the determination of the transfer function is simplified. For the sake of clarity we can take for an example the case where the input is a single rectangular pulse of height U_0 and d+1 samples in duration. The z transform of this input is

$$U(z) = U_0 \sum_{j=0}^{d} z^{-j} \tag{5.8}$$

Applying this input to a linear transfer function yields the output signal Y(z)

$$Y(z) = G(z)U(z) = U(z) \sum_{i=1}^{n} \frac{R_i}{z-\lambda_i} \tag{5.9}$$

where the right hand part of this expression contains the transfer function in partial fraction form. The R_i's in (5.9) are transfer function residues which can be mathematically determined from the output signal residues, which are the $\overline{R}_i$'s, in (5.7). Combining equation (5.9) with the previous expression for U(z) we can get

$$Y(z) = U_0 \sum_{j=0}^{j=d} z^{-j} \sum_{i=1}^{n} \frac{R_i}{z - \lambda_i} \tag{5.10}$$

The inverse z transform gives the following sampled output signal

$$y(k) = U_0 \sum_{j=0}^{d} \sum_{i=1}^{n} R_i (\lambda_i)^{(k-j-1)} \quad u_s(k-j-1) \tag{5.11}$$

where u_s is the unit step function. This expression is actually a discrete time convolution of the sampled input signal with the system impulse response. When the input is no longer being applied to the system then the output can be expressed as

$$y(k) = U_0 \sum_{i=1}^{n} R_i (\lambda_i)^k \sum_{j=0}^{d} (\lambda_i)^{-(j+1)} \qquad for \quad k > d \tag{5.12}$$

The last summation in this expression can be rearranged to yield

$$y(k) = U_0 \sum_{i=1}^{n} R_i (\lambda_i)^k \left(\frac{-1 + \lambda_i^{-(d+1)}}{1 - \lambda_i} \right) = \sum_{i=1}^{n} \overline{R}_i \lambda_i^k \qquad for \quad k > d \tag{5.13}$$

This last expression illustrates the difference between transfer function residues R_i in (5.12) and the signal residues $\overline{R}_i$ given by (5.7). This relationship for this case is

$$R_i = \frac{\overline{R}_i}{U_0}\left(\frac{-1+\lambda_i}{1-\lambda_i^{-(d+1)}}\right) \tag{5.14}$$

Thus by using the information obtained from the ring-down signal description obtained by applying the basic Prony method and utilizing known information about the input signal, it is possible to solve for the system transfer function between the input signal and the output of the system. More general approaches to obtaining transfer functions are presented in [22,61]. These references also describe a procedure for obtaining transfer functions when the system is not initially at rest. This basic Prony algorithm has several modifications which have not yet been thoroughly tested.

The transfer function can be checked for accuracy by applying the same input to this model and calculating the resulting output and comparing this signal to the actual system output. If the calculated system response is very close to that of the actual system response, this is a positive indication that the Prony method has successfully located the dominant modes of the system and the correct damping terms for these modes. It is also a good indication that the transfer function is accurate. In some cases, however, even if the output signals are nearly identical, the transfer function may not accurately predict the effect of feedback on the actual system poles. Further checking is needed to verify the adequacy of a fitted transfer function especially if the intended use is for control systems design. An illustration of this problem is presented in case 3 in the following section.

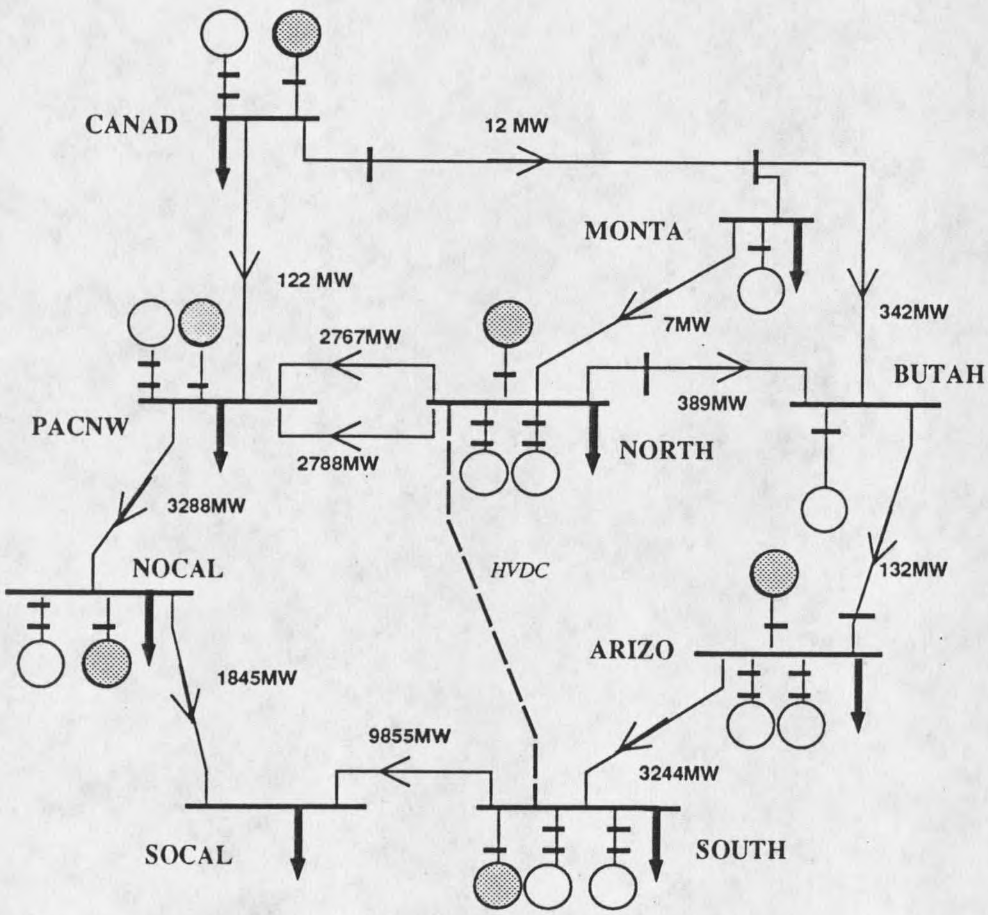

Figure 5.2. A one line diagram of the test system under consideration.

Application Examples

The following examples illustrate several applications of a TFI method to a variety of situations in a test system. The test system is a low-order power system model as described in Chapter 2. The system contains interarea modes which are in the same frequency range as those which have been documented in the western North American power system. A one-line diagram of the test system is shown in Figure 5.2. The major buses in the system are all 500 KV except for the bus BUTAH which is 345 KV. This system contains several classical machines which are indicated by cross-hatching and detailed two axis machines which do not have the cross-hatching. The classical machines have very large power generation and large inertia, and their parameters were chosen to produce the interarea modes of interest. A detailed description of the development of this test system is given in Chapter 2.

Case 1

The first example of an application of a TFI method is taken from attempts to validate software. The software in this case is the ETMSP program. The objective is to determine if a governor model is being simulated properly. To determine this a rectangular probing pulse is applied to the reference input on the governor for a particular machine. A block diagram for the governor is shown in Figure 5.3. The frequency deviation input to the governor controls was disabled in order that the closed-loop system eigenvalues would not contain any electromechanical modes. This allows the closed-loop system modes to be calculated entirely from the diagram of Figure 5.3. The P_{gv} signal at the output of the governor was recorded and this information was processed by the Prony

TFI method. A comparison of the closed-loop eigenvalues taken from the block diagram and those obtained from the TFI algorithm is shown in Table 5.1. All of the closed-loop eigenvalues are real in this case. The comparison indicates that under linear and noise free conditions the Prony algorithm can identify transfer function poles and residues very accurately. It appears that the estimates of the poles are slightly more accurate than those of residues. This situation has been observed in many test cases.

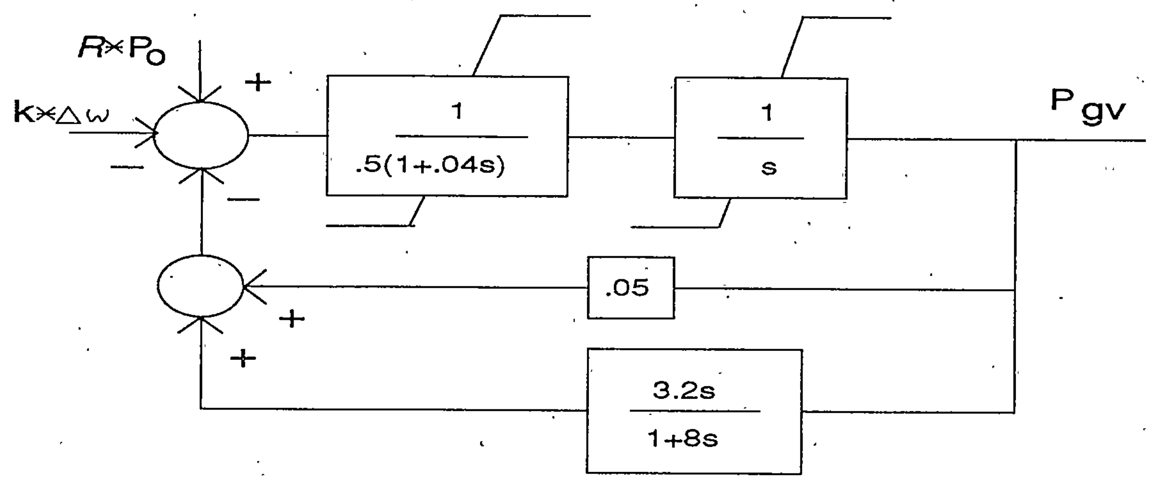

Figure 5.3. Governor block diagram.

Parameters identified using Prony TFI		Parameters from Block Diagram (closed-loop)	
Eigenvalue	Residue	Eigenvalue	Residue
-.0123	0.231	-.0123	0.225
-1.052	1.998	-1.052	1.937
-23.7	-2.654	-24.1	-2.163

Table 5.1. Comparison of identified eigenvalues and residues versus the parameters expected from the closed-loop block diagram.

__Case 2__

In this example an SVC has been placed at the bus SOCAL in the test system of Figure 5.2. The SVC has a voltage control loop for maintaining the voltage at this bus. The TFI method is then applied to determine what signals and what interarea modes might be strongly affected by a second feedback loop. In this case we are using the relative phase angle between bus SOCAL and PACNW as the output of the system. A block diagram of the SVC is shown in Figure 5.4.

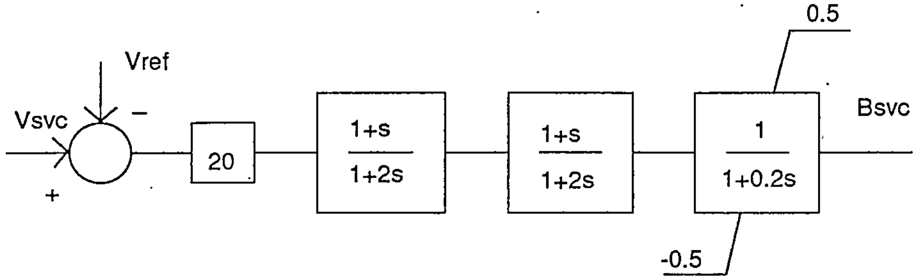

Figure 5.4. SVC block diagram.

The TFI study is carried out by applying a rectangular probing pulse to the reference input on the SVC. The output is the change from steady state of the relative phase angle between the two parts. These signals are then used in the Prony TFI method to determine the transfer function between the input and this output signal. There are two tests which are used to determine the validity of the fitted transfer function model. The first one is to apply the same input signal to the transfer function model and see how the output signal generated compares to the output which was used to determine the fit. A comparison of the actual output and the identified model output is shown in Figure 5.5.

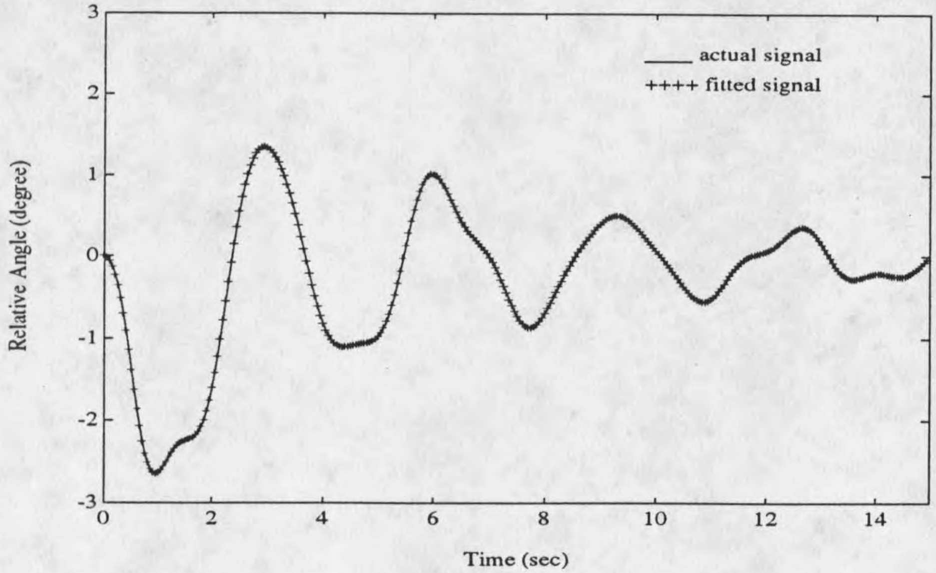

Figure 5.5. A plot of the actual output signal and the fitted output signal. The signal is the change from the steady state value of the relative angle between two buses.

The curves in Figure 5.5 indicate a very good fit since the actual and fitted model output are lying on top of each other. This is an initial indication that the identified transfer function is accurate. However, experience has shown that a good match to the signal cannot always be trusted as an indicator that the identified transfer function is valid. Another more reliable test is what we will call the root locus test (RL test). The basic idea here is that once the open-loop transfer function is identified then a root locus analysis is done on this transfer function to determine how the closed-loop system poles are expected to move as the feedback loop gain is varied. Next the loop on the SVC is closed with some gain. A pulse is again applied to the SVC with the loop closed and the closed-loop poles are identified. If the identified closed-loop poles agree with the pole locations in the root locus, then the identified open-loop transfer function model is accepted as a valid transfer function for the system.

The results here are illustrated by computer simulation of the above power system, but this same method could also be applied to measurement from an actual power system. The root locus in Figure 5.6 was run using feedback gains from 0 to 0.2 in 20 increments of 0.01. The amount of pole movement even for small gains illustrates that at least two of the interarea modes are very sensitive to feedback gain. A comparison between the root locus poles for a gain of 0.01 and the closed-loop system identified poles is shown in Table 5.2. These results indicate that the open-loop identified transfer function is accurate, at least for feedback gains in the range from 0.01 to about 0.05.

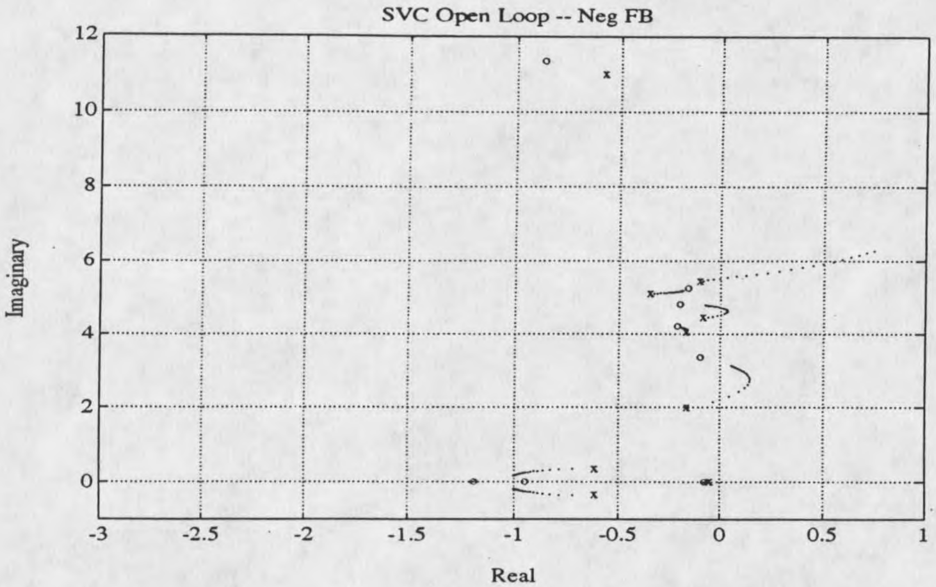

Figure 5.6. Root Locus of identified open-loop transfer function.

The data in Table 5.2 shows that there is good agreement between the open-loop transfer function root locus and the closed-loop identified eigenvalues. This includes one eigenvalue which has moved from -.161 + j1.988 in the open-loop to +0.06 + j2.34 in the closed-loop system with gain of 0.02. With high gains the system becomes more unstable and saturations occur within the SVC. The frequencies of unstable oscillations however are accurately predicted by the root locus of the identified open-loop transfer function.

closed-loop identified eigenvalues (gain=0.01)		closed-loop eigenvalues from root locus (gain=0.01)
-0.0215 ± j2.181		-0.0323 ± j2.161
-0.0625 ± j4.440		-0.0614 ± j4.480
-0.0939 ± j5.474		-0.0843 ± j5.469
-0.1612 ± j4.107		-0.1641 ± j4.083
closed-loop identified eigenvalues (gain=0.02)		closed-loop eigenvalues from root locus (gain=0.02)
+0.0602 ± j2.343		+0.0443 ± j2.309
-0.0440 ± j4.550		-0.0389 ± j4.498
-0.0550 ± j5.500		-0.0623 ± j5.492
-0.3203 ± j3.997		-0.1621 ± j4.100

Table 5.2. Comparison of closed-loop eigenvalues for gains of 0.01 and 0.02. Eigenvalues in the right column are obtained from the root locus of identified open-loop transfer function. Those on the left are identified from a closed-loop simulation.

Case 3

The third example is an extension of the previous case to illustrate some of the difficulties inherent in fitting transfer functions. Figure 5.7 illustrates a Prony transfer function fit on the same case above except for positive feedback and a gain of 0.05. This positive feedback stabilizes the interarea modes of the system but causes one real eigenvalue to shift to an unstable value of +0.6. When a simulation case was run at this feedback gain the system was found to be stable. Closer analysis of the situation showed that about 1 second after the probing input was applied the SVC hits its upper limit and stays there for the duration of the simulation. Due to this saturation condition the output signal being analyzed must be considered nonlinear. A transfer function fit of the closed-loop system was run and the output signal is compared to the identified model signal in Figure 5.7. The two curves lie on top of each other indicating a good signal fit. This example illustrates a case where a signal that is produced by a nonlinearity can still be accurately fitted with linear parameters using a TFI algorithm.

The important implication here is that, even though the fitted closed-loop model output signal agrees very well with the actual output signal, the resulting transfer function is not legitimate for control purposes because of the saturation occurring within the SVC. This is a situation where although the Prony model looks good when comparing the fitted and actual output signal, some of the eigenvalues of the Prony identification are not valid due to nonlinearities (saturation) under the conditions of operation. In this particular situation the open-loop transfer function model predicts effects of feedback accurately only until the system begins to go unstable and then the model is no longer valid. It appears that the saturation within the SVC is preventing the system from going unstable with higher feedback gains. By contrast when the system is forced into unstable regions using negative feedback gains, the system exhibits growing oscillations until a limit cycle is reached.

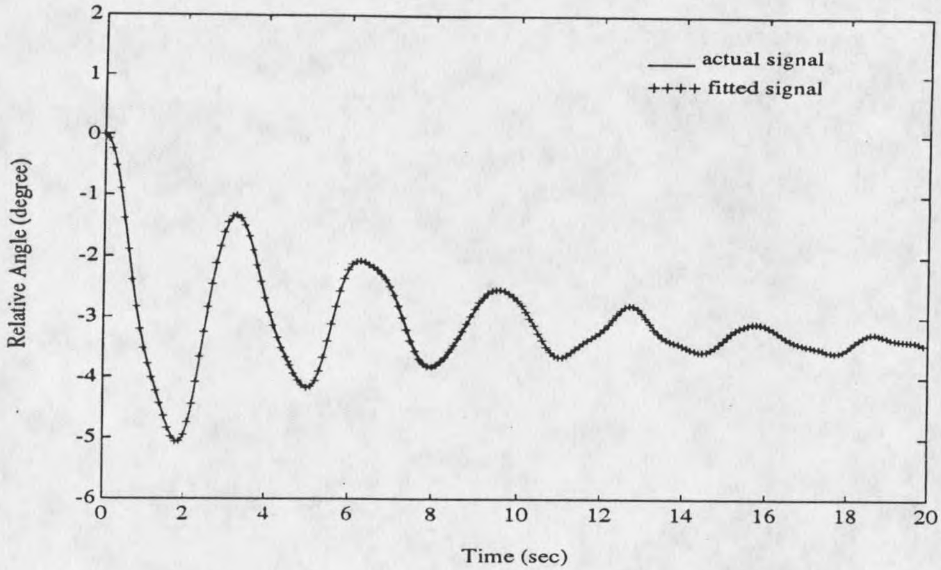

Figure 5.7. Fit of relative angle affected by saturation of SVC unit.

Case 4

To further verify the validity of transfer function identification techniques, a comparison was made between the eigenvalues obtained from the Prony TFI method and the eigenvalues obtained from PEALS [52]. The system used in both cases was the same 46-bus system described earlier. Table 5.3 compares the identified eigenvalues from the open-loop SVC probing to the eigenvalues obtained from PEALS using the same stability case data. In general it can be difficult to compare eigenvalues between these two approaches. The difficulty lies in the fact that the TFI methods give system eigenvalues that can be seen from a particular input and output signal in the system, but the eigenvalues from PEALS are not associated with any particular reference input of the system. PEALS searches in different frequency ranges to find eigenvalues. For a large system PEALS will find some eigenvalues in virtually any part of the s-plane and there will often be several eigenvalues near each other which are associated with different parts of the system. As an example PEALS may find several eigenvalues with a frequency near 0.6 Hz, where only one of these may be significant from a particular input/output interaction within the system: hence, the TFI method will only report one eigenvalue. The relatively small number of states in the test system used here alleviates this problem somewhat. As the table indicates, there is generally good agreement between the results of the two methods especially when considering the fundamentally different approaches to arriving at the results.

Eigenvalues obtained from PEALS		Eigenvalues obtained by Prony TFI
$-0.2092 \pm j1.999$		$-0.1619 \pm j1.988$
$-0.1173 \pm j4.644$		$-0.0841 \pm j4.465$
$-0.1107 \pm j5.475$		$-0.1017 \pm j5.449$
$-0.1292 \pm j4.205$		$-0.1684 \pm j4.064$

Table 5.3. Comparison of interarea modes of the test system.

Discussion of Results

An overview of two different transfer function identification methods has been presented. One is the direct ARMA TFI method and the other is the Prony TFI method. Both methods have been found to be effective in transfer function identification in power systems and both methods have their advantages and disadvantages. The advantage of the Prony method is the numerical robustness obtained with the parallel formulation of the equations. The advantage of the direct ARMA method is its simplicity. A key feature of both of these methods is the shape of the data matrix needed for obtaining good fits. This usually means that both methods will find accurate values for the transfer function along with many extraneous values which may need to be removed. A feature that is often useful in identifying the extraneous values is the fact that the imaginary part of the extraneous eigenvalue will often occur at frequencies that are known to be absent from the recorded signal.

Several examples of applications of TFI methods have been presented. The first one applied a TFI method to a simulated signal to verify the block diagram that a stability program was supposed to implement. The second example applied a Prony TFI method to a secondary damping control loop on an SVC. In this second case the actual transfer function was not known. A root locus test was performed to illustrate a method for validating transfer functions obtained from identification methods. In a third case the eigenvalues obtained from a fit to a nonlinear signal were presented. In the last case, eigenvalues from the Prony TFI method were compared against those obtained from PEALS. The results between these entirely different approaches show good agreement which indicates that both methods can be valid ways of obtaining key parameters of power system dynamics.

CHAPTER 6

CLOSED-LOOP IDENTIFICATION AND TUNING FOR DAMPING OF INTERAREA MODES

This chapter presents recent results on investigations into the use of least-squares transfer function identification methods for the design of controllers to enhance the damping of interarea oscillations in multimachine power systems. As mentioned in Chapter 5 numerically identified transfer function models can be very useful in a variety of power system applications [27-31]. Recent papers in transfer function identification have reported significant advantages on identifying the plant from closed-loop operation rather than from the open-loop state [32-38]. The purpose of this chapter is to report on recent findings using closed-loop identification methods to develop system models for designing controllers to enhance the damping of interarea oscillations. An iterative closed-loop identification method is used to find a linear model for the power system in the neighborhood of a given operating point. The identified model is a nonminimum-phase (NMP) power system model, and therefore an advanced robust linear quadratic Gaussian controller design with loop transfer recovery (LQG/LTR) based on a partial loop recovery technique [41-43] is used to dampen low-frequency oscillations. The details on LQG/LTR control strategy for NMP plant models is provided in the next chapter.

The device being used to dampen oscillations is a static var compensator (SVC) placed in a low-order nonlinear power system model.

Nonlinear power system simulations are used to illustrate the effectiveness of using models identified on the basis of closed-loop response data (closed-loop identified models) compared to models identified on the basis of open-loop response data (open-loop identified models).

Closed-Loop Identification

Power systems like many physical systems contain significant nonlinearities. In power systems these include trigonometric and exponential functions in addition to the more common saturation functions.

In spite of the nonlinearities, numerically identified transfer function models can be very useful in a variety of power system applications. Any linear model of a power system has a limited range of gains and operating conditions over which it will accurately portray the dynamic behavior of the system. Due to this circumstance it appears reasonable to expect that the power system model which is the most appropriate in linear controller design is the one fitted to the conditions nearest to the actual controller operating environment.

In designing a controller for an SVC to enhance the damping of interarea oscillations, the linearized model used in the design can have a significant impact upon the effectiveness of the controller when it is applied to the actual system. Linearized models that result from the analysis of nonlinear system data depend on the characteristics of the applied input signals. As mentioned in the previous chapter, an open-loop identified linear model of an SVC may not predict the closed-loop system behavior accurately except at very small loop gains.

Recent papers on iterative identification and control design have recommended using input-output data collected from the plant during closed-loop operation in order to obtain improved plant models. The iterative scheme involves identification and controller design coupled together in two phases to seek an improvement of some performance criterion with each iteration. A primary aim of the iterative approach is to focus the identification phase more appropriately on the intended operating condition and thereby incorporate robustness into the controller design.

In the first phase of the identification method, a rectangular pulse is applied at the reference input signal and the output response signal is recorded. Using the Prony identification method [27,30 and 35], a least squares fitted transfer function for the plant G is obtained. Figure 6.1 contains a block diagram illustrating the SVC controls and associated plant, G, to be identified. When the plant is in the open-loop state this identified transfer function will be called G_1. The model G_1 is then used to design a controller to improve damping of interarea modes. The controller has a transfer function of H as illustrated in Figure 6.1. This transfer function is designated as H_1 indicating it has been designed based on the plant model G_1.

Once the loop is closed with H_1 operating as the compensator, we again identify the plant by applying a pulse input to the reference signal and measuring the response signal. In this way the closed-loop system model T, where

$$T = \frac{G}{1 + GH} \tag{6.1}$$

is the model actually obtained. In this relation H is know, and G can be obtained as

$$G = \frac{T}{1 - TH} \tag{6.2}$$

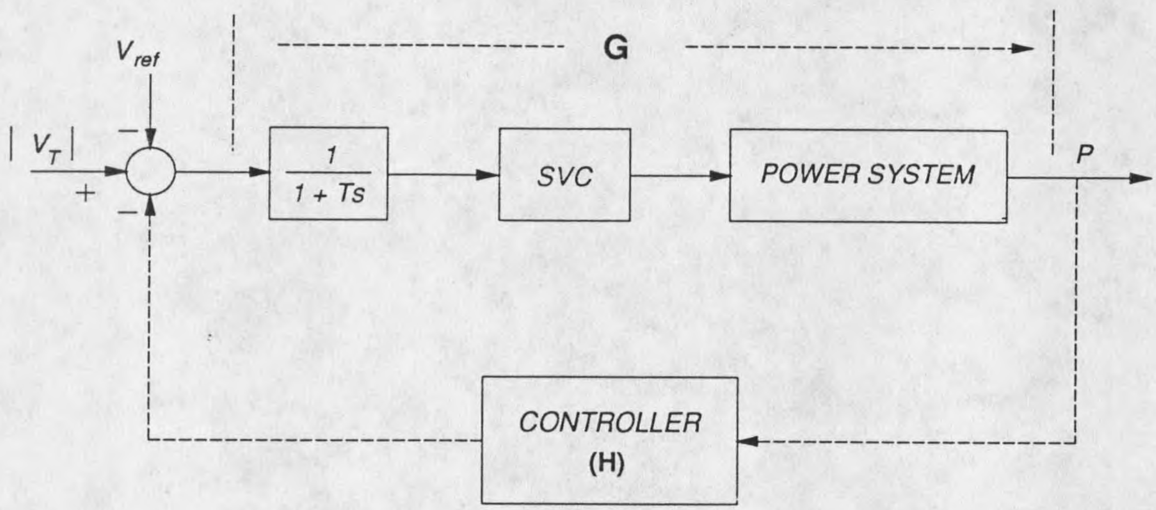

Figure 6.1. A block diagram of SVC controls and associated plant.

The plant model obtained in this way is denoted by G_2. Once G_2 is obtained another controller is designed using this updated plant model, and this iterative procedure is continued until further improvement is insignificant.

Figure 6.2 shows the system diagram which is used for designing the controller. The controller is a linear quadratic Gaussian controller based on partial loop transfer recovery (LQG/LTR) which will be explained in the following chapter.

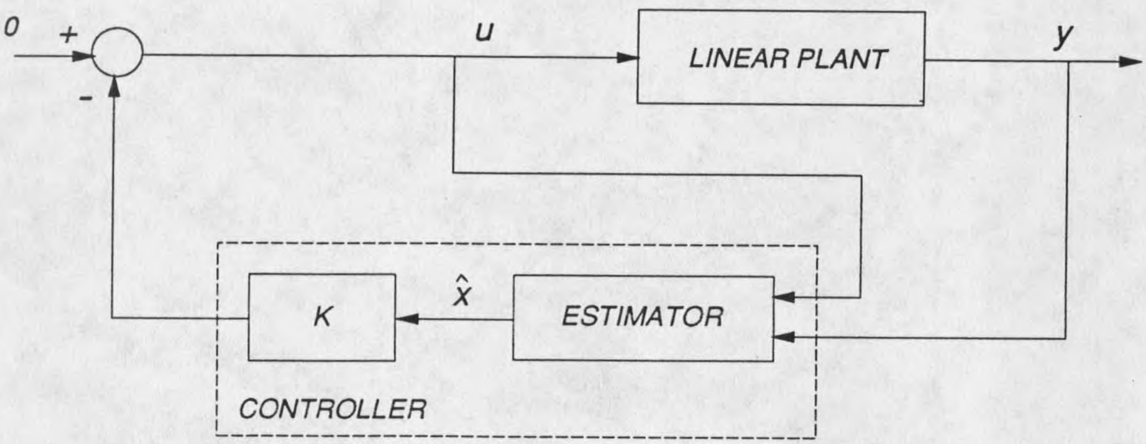

Figure 6.2. System diagram of linear plant and controller.

Test System Description

The power system example described here consists of an LQG/LTR controller applied to a static var compensator (SVC) in a power system simulation. The nonlinear power system model under consideration is the low-order model which was described in Chapter 2 (Figure 5.2).

The SVC is located at the bus SOCAL. The primary purpose of the SVC is local bus voltage control. The SVC acts effectively as a thyristor controlled variable shunt susceptance in the power system network. In response to an input signal, the shunt susceptance can be changed very rapidly to modify the reactive power and voltage magnitude at the local bus. Interarea damping control is added as a supplementary control loop which utilizes line power between NOCAL and SOCAL as the input signal. This signal was selected as the controller input because it has good controllability and observability characteristics for several interarea modes.

Simulation Results

Several different tests are described in this section. For the cases that involve pulse reference inputs, a rectangular pulse is applied to the reference input of the SVC at the SOCAL bus; for the cases that involve a three phase fault, the fault is applied at NORTH; the fault is initiated 2 seconds after the start of the simulation. For the cases that involve a change in operating point, a line is removed between CANAD and PACNW buses.

Comparison of Open-Loop and Closed-Loop Models

In the following tests two different levels of rectangular pulse (0.3 and 2.5) are applied to the power system, and open-loop and closed-loop identified models are compared. First, a low level rectangular pulse (0.3) is applied at the reference input signal, and the output response signal (line power between NOCAL and SOCAL) is recorded. Using the prony identification method [27], a least squares fitted transfer function for the plant is obtained. Figures 6.3 and 6.4 show the power system output and the identified (linear) plant model outputs for two different model orders.

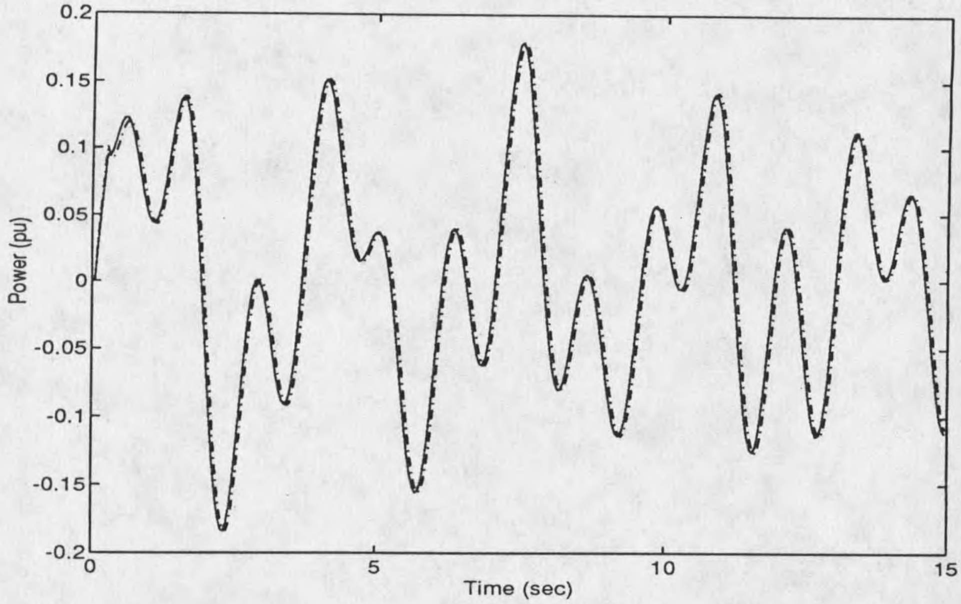

Figure 6.3. Open-loop power system output (solid line) and open-loop identified plant model output (dash dot),tenth-order.

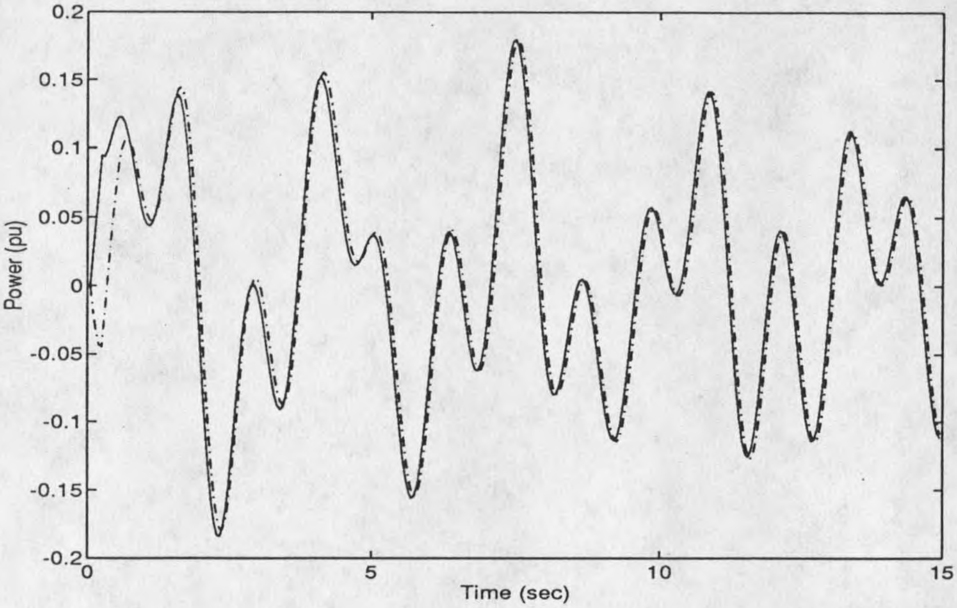

Figure 6.4. Open-loop power system output (solid line) and open-loop identified plant model, G_1, output (dash dot), sixth-order.

An LQG/LTR (trial approach as will be explained in the following chapter) controller design based on the sixth-order open-loop identified plant model (G_1) is applied to the power system . Figure 6.5 compares the power system output and the linear model output with control, and Figure 6.6 compares pole locations of open-loop (G_1) and closed-loop (G_2) identified plant models. As these figures show, for this level of probing the power system exhibits linear characteristics, and the open-loop and closed-loop identified plants models are similar (in Figure 6.6 the two models have essentially the same three main modes, and the closed-loop identified plant model has some neighboring poles and zeros that effectively cancel).

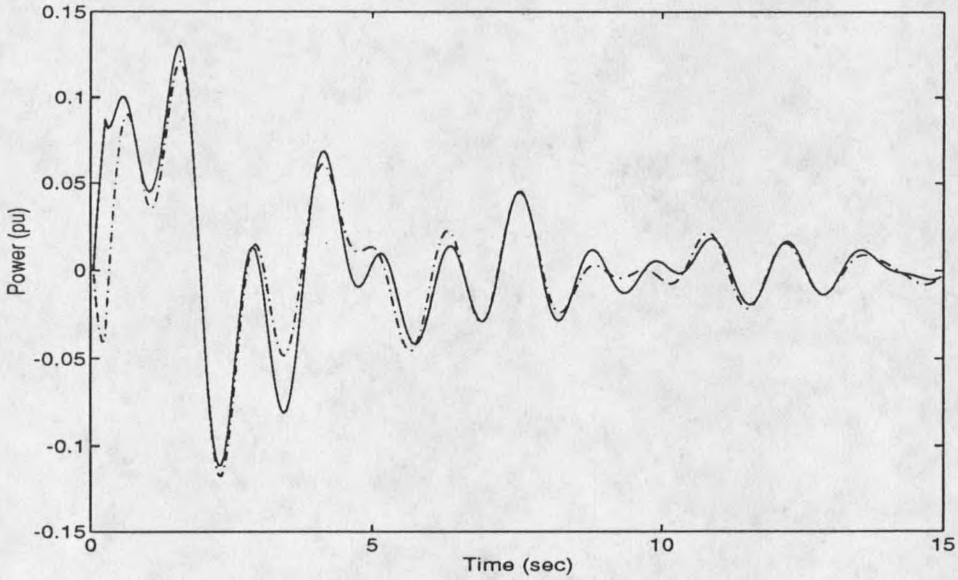

Figure 6.5. Power system output (solid line) and identified plant model (G_1) output (dash dot) with control.

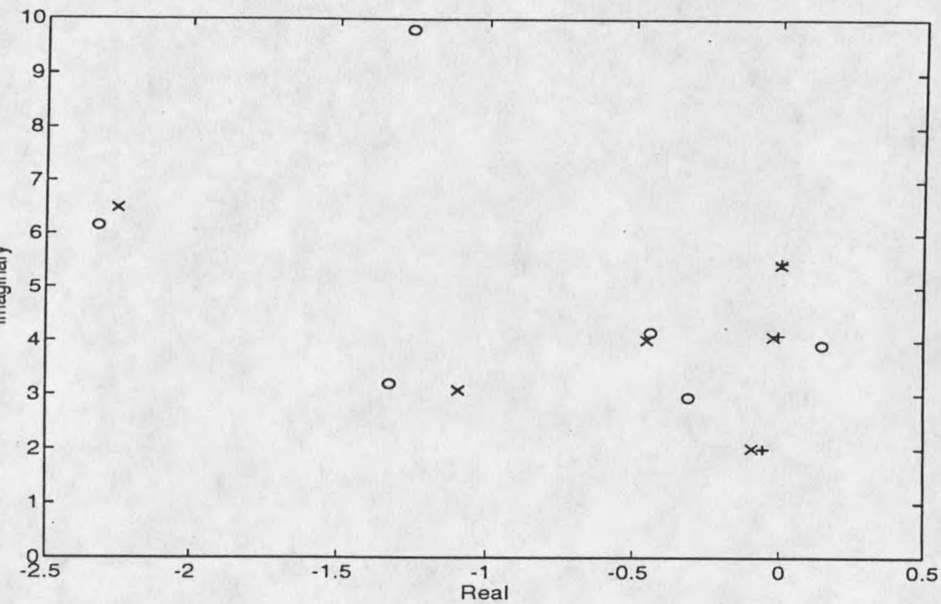

Figure 6.6. Open-loop (G_1) poles (+) versus closed-loop (G_2) poles (x) and closed-loop zeros (o).

A similar test with a high level rectangular pulse (2.5) illustrates the nonlinearity of the power system and the difference between the new open-loop (G_1) and the new closed-loop (G_2) identified plant models (Figures 6.7 and 6.8). The power system output in Figure 6.7 is more damped than the output of the controlled open-loop identified plant model; this correlates with the fact that the dominant closed-loop identified plant model poles of Figure 6.8 are farther to the left in the s-plane than the dominant poles of the corresponding open-loop identified plant model. By using this updated identified plant model (G_2) in redesigning the controller, the controlled output of the power system is seen (Figure 6.9) to be closer to that of the controlled output of the new linear plant model. By repeating this iterative procedure, the power system output shows better agreement with the controlled output of the new linear plant model (G_3) as illustrated in Figure 6.10. Figure 6.11 compares power system outputs with controllers designed based on plant models G_3 and G_1 with higher control gain.

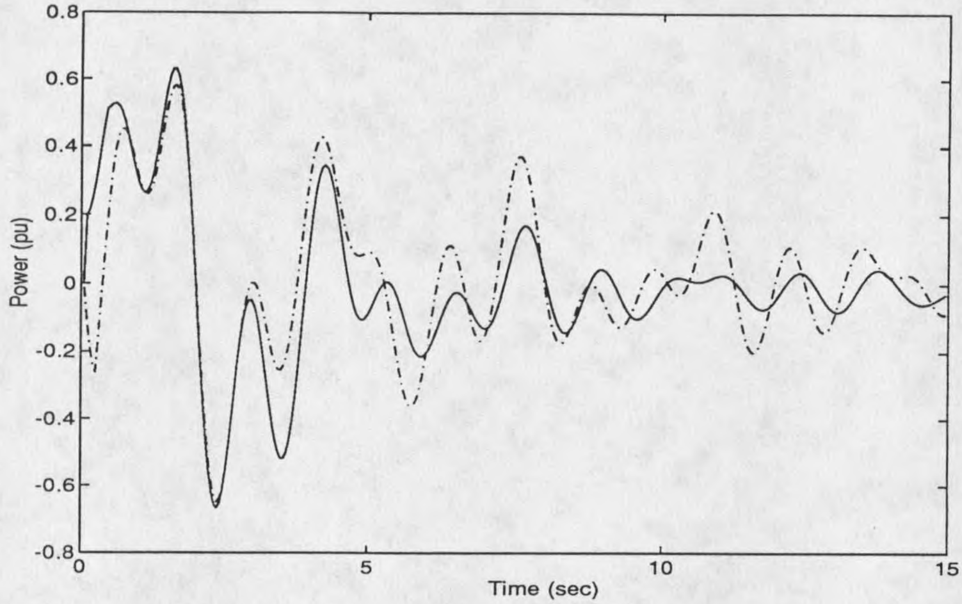

Figure 6.7. Power system output (solid line) and identified model (G_1) output (dash dot), with high level rectangular pulse and with control.

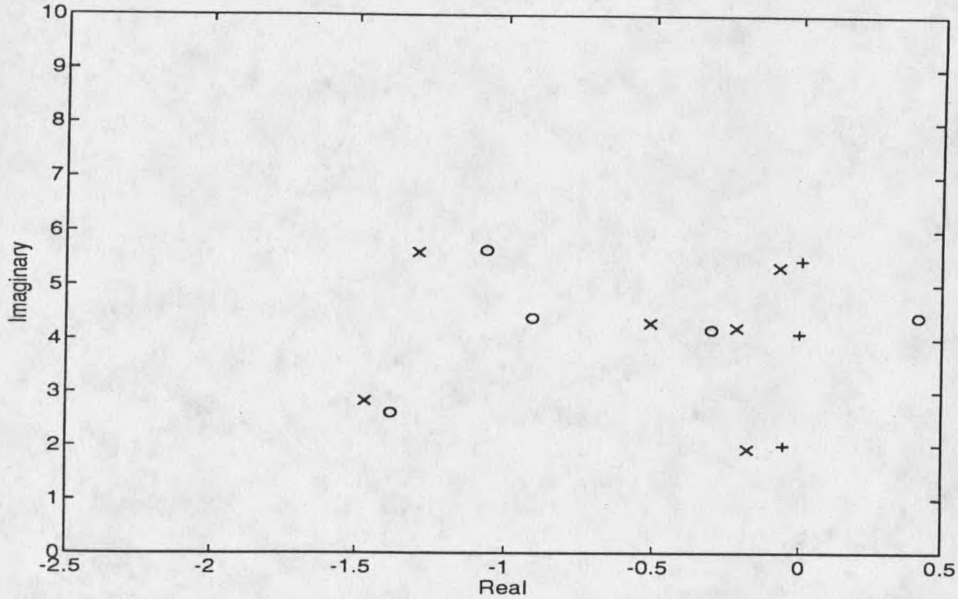

Figure 6.8. Open-loop (G_1) poles (+) versus closed-loop (G_2) poles (x) and closed-loop zeros (o) with high level rectangular pulse.

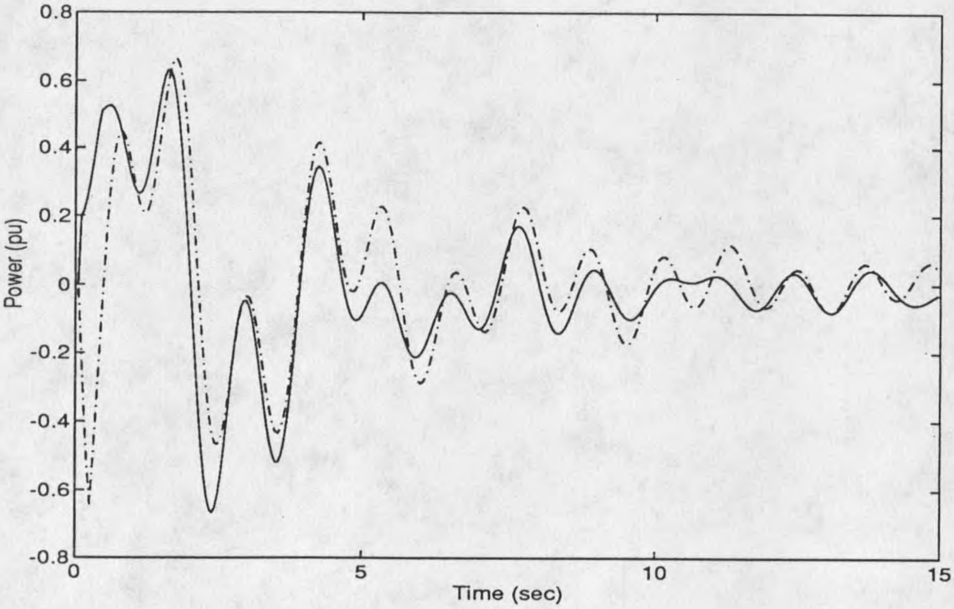

Figure 6.9. Power system output (solid line) and identified plant model (G_2) output (dash dot), with high level rectangular pulse and with control.

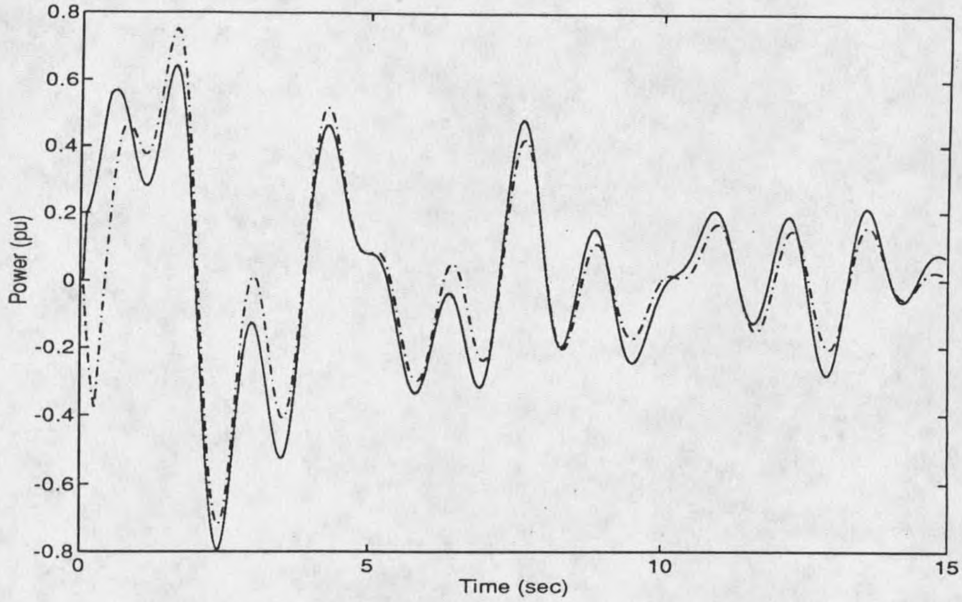

Figure 6.10. Power system output (solid line) and identified plant model (G_3) output (dash dot), with high level rectangular pulse and with control.

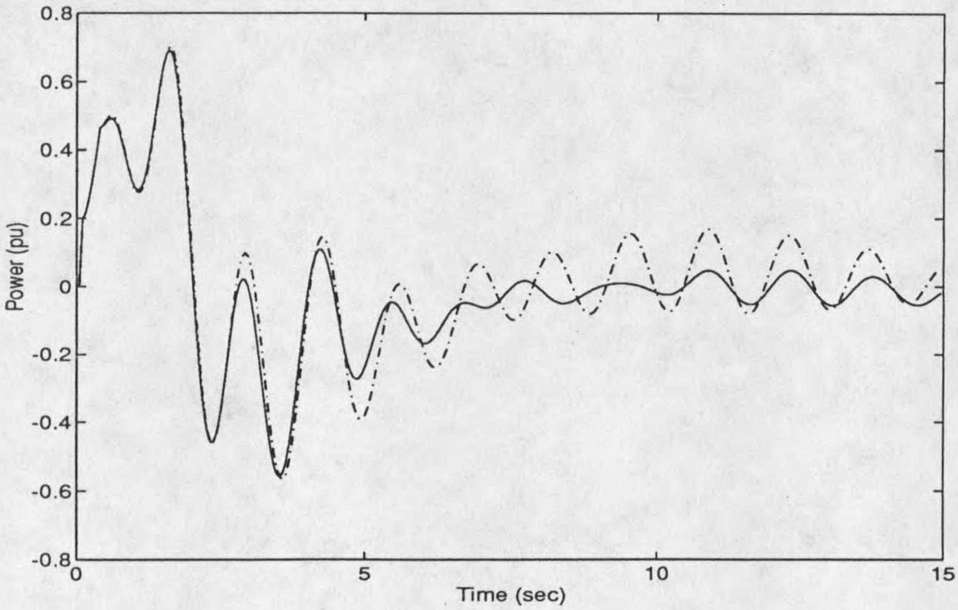

Figure 6.11. Power system outputs with controller designed based on plant models G_3 (solid line) and G_1 (dash dot) with higher control gain.

Closed-Loop Control in the Presence of a Fault

The controller design based on closed-loop model (G_3) is applied to the power system, and the resulting control performance is illustrated for different fault conditions.

Figures 6.12 and 6.13 illustrate the effectiveness of the controller in damping oscillations which are observed in the relative angle (between SOUTH and CANAD), and line power (between CANAD and PACNW buses); the disturbance in these figures is a three-phase fault (the fault is applied at NORTH).

In changing the operating point by removing a line (a line is removed between CANAD and PACNW buses), the controller supplies considerable damping improvement over the no control case. This is illustrated in Figures 6.14 and 6.15.

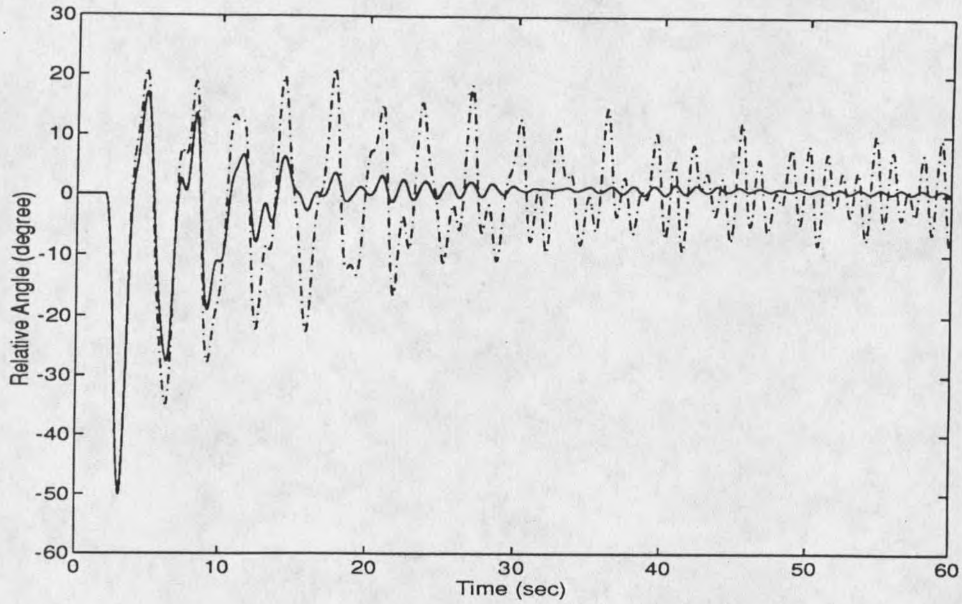

Figure 6.12. Relative angle (degree), solid line with control and dash dot without control; three phase fault.

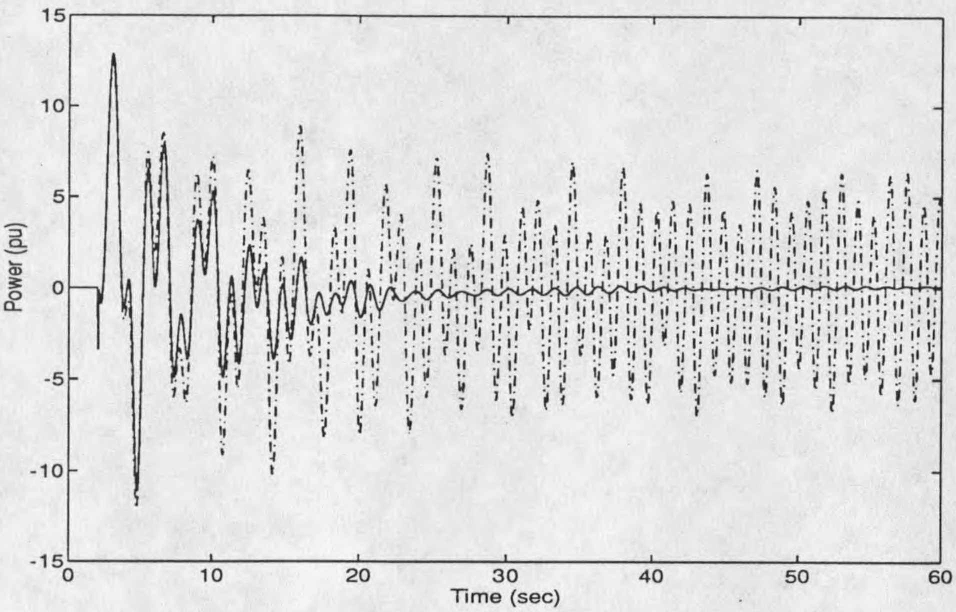

Figure 6.13. Line power, solid line with control and dash dot without control; three phase fault.

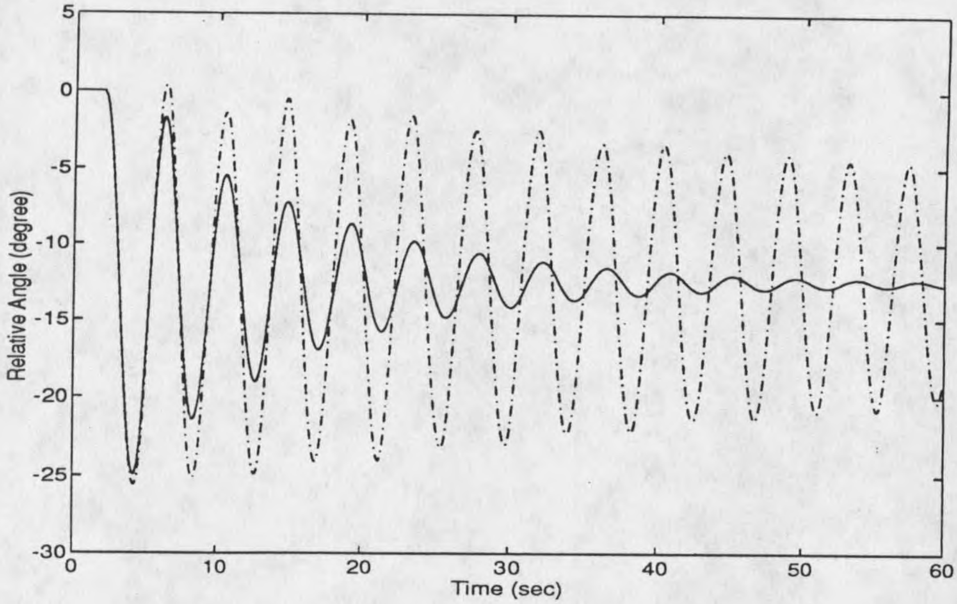

Figure 6.14. Relative angle (degree), solid line with control and dash dot without control; line removed.

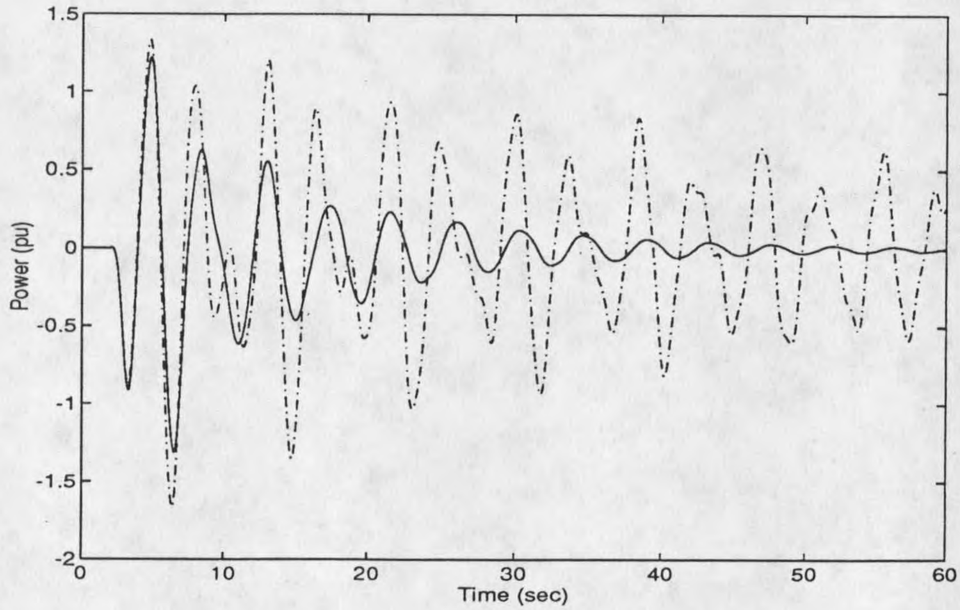

Figure 6.15. Line power, solid line with control and dash dot without control; line removed.

Discussion of Results

Controllers are designed to dampen oscillations in a low-order nonlinear power system model which has dynamic characteristics similar to the western North American power system. An iterative closed-loop identification method is applied to obtain linear models of the power system for use in controller design. Although the control is applied to a static var compensator (SVC) in the system, the design process presented can be readily applied to many other power system devices. Two different probing levels are used for model identification to demonstrate both the underlying nonlinear nature of the power system and the effectiveness of using the closed-loop iterative identification method in the case of large probing inputs. The identified plant models contain nonminimum phase zeros, and controllers are designed using linear quadratic Gaussian theory with partial loop transfer recovery (LQG/LTR). Nonlinear power system simulations are used to illustrate the effectiveness of the resulting control in damping system oscillations even with a change in operating point.

CHAPTER 7

ROBUSTNESS OF LQG/LTR CONTROL STRATEGY

In this chapter an advanced robust LQG/LTR controller is designed to dampen low-frequency oscillations in a large interconnected power system. The plant identified model contains nonminimum phase zeros; two methods of partial loop recovery for NMP plants are described and compared [41,42].

Robustness of the control is examined relative to realistic load noise and faults that change the system operating point. The LQG/LTR controller is used to control a static var compensator (SVC) placed in the low-order power system model with interarea modes similar to those of the western North American power system, as described in Chapter 2 and [31], to dampen the system low-frequency oscillations.

Nonlinear power system simulations are used to illustrate the effectiveness of the robust controller designs based on the amount of loop transfer recovery that is possible for the NMP plant model.

LQG/LTR Control Strategy

Linear quadratic Gaussian (LQG) control design is an eminent method of state feedback design which combines linear quadratic (LQ) control theory to compute an optimal gain vector and Kalman filter theory to generate state estimates. Consider a stochastic plant model, with state equation

$$x(k+1) = Fx(k) + Gu(k) + Mv \tag{7.1a}$$

and output equation

$$y(k) = Hx(k) + w \tag{7.1b}$$

where x is the discrete-time state vector, u is the control vector, y is the output vector, v and w are noise vectors, and F, G, H, and M are constant system matrices of appropriate dimension. In Figure 7.1 the block labeled *linear plant* designates the relationship between u and y of (7.1), where for ease of presentation the noise signals v and w are not shown in the figure.

The estimator of Figure 7.1 is characterized by

$$\hat{x}(k+1) = F\hat{x}(k) + Gu(k) + K_e(y - H\hat{x}(k)) \tag{7.2}$$

where $\hat{x}$ is the estimate of x and K_e is the estimator gain matrix. Also shown in Figure 7.1 is the formation of the control signal u in terms of the controller gain matrix K and the state estimate $\hat{x}$,

$$u(k) = -K\hat{x}(k) \tag{7.3}$$

It is known that guaranteed robustness against perturbations at the plant input of a state feedback design can deteriorate with the use of an observer [2]. The robustness properties are determined by the loop gain (LG) and the sensitivity function at the input of the plant. For good disturbance suppression the input sensitivity function $(1+LG)^{-1}$ should be low. Alternatively the return difference $(1+LG)$ should be greater than one in the frequency range of interest (return difference inequality).

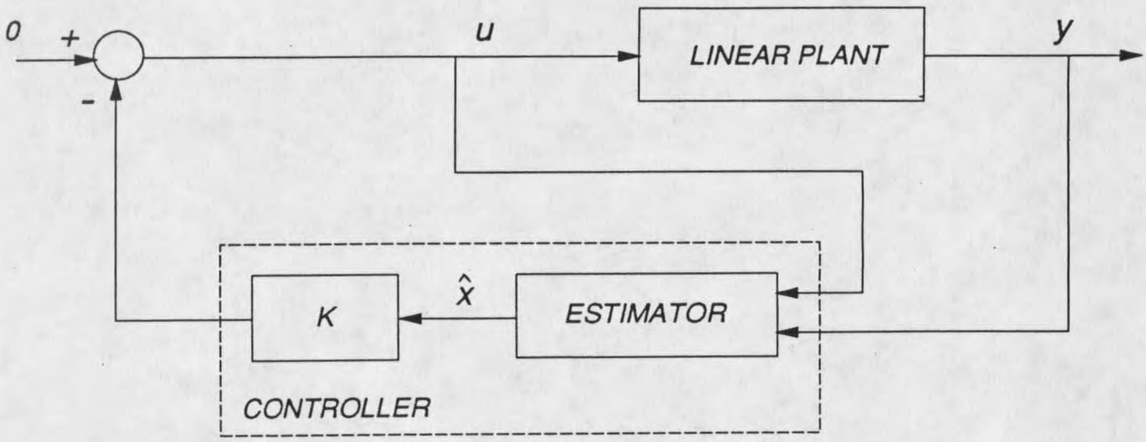

Figure 7.1. System diagram of an estimator/regulator.

In Figure 7.1 the loop gain depends on where the loop is opened. If the loop is opened just after the summing function then the loop gain is simply $-K(zI-F)^{-1}G$, and is identical to that with full state feedback (no observer). However, to analyze input robustness it is usual to open the loop at the plant input. Then the loop gain is

$$W_{OL}(z) = [-K(zI - F + GK + K_e H)^{-1}(K_e)]\,[H(zI - F)^{-1}G] \tag{7.4}$$

which is not the same as that which is obtained with full state feedback. Thus with estimated state feedback we cannot expect the return difference inequality to hold, with its associated guaranteed robustness properties [41].

Loop Recovery

Linear quadratic Gaussian control with loop transfer recovery (LQG/LTR), was developed by Doyle and Stein and discussed further by Stein and Athans [41]. With this method, the loop gain for the LQG design is recovered to that of the LQ design under appropriate conditions, and there is recovery in associated robustness properties.

Starting with a nominal stochastic design that lacks the robustness properties of the LQ design, fictitious noise is added to the plant input model in the estimator (Kalman filter) design stage, affecting the resulting values in K_e. This fictitious noise represents plant variations, uncertainties, and unmodeled dynamics. When the plant is minimum phase (no right half-plane zeros), this technique leads to "loop recovery" because as the magnitude of the fictitious noise increases, the loop gain and its associated input sensitivity function approach those of the state feedback design. However, if the plant is non-minimum phase (NMP), the LTR technique cannot recover the state feedback arbitrarily well.

The design procedure can be summarized as follows. First, using the identified NMP plant model and temporarily assuming full state feedback (SF design), select LQ performance weights to obtain a balance between added damping and sensitivity reduction. Second, design a Kalman filter as an estimator to be used with the controller to achieve a state estimate feedback (SEF) design; tune the filter design by raising the level of input process noise (fictitious noise) until the sensitivity at the plant input for the SEF design is similar to that of the SF design in the frequency range of interest. Because of the added fictitious noise in the filter design, the robustness of the final control is improved but its nominal performance is reduced; there is a trade-off in the design between performance loss at design center and robustness gain. As discussed in the *Factored Plant Model Approach* subsection below, the particular state-space representation used in the above design can have an impact on the results.

Loop Transfer Recovery with NMP Zeros

It is known that a system with NMP zeros is constrained by those zeros [43]; i.e., sensitivity reduction (loop recovery) achieved in one frequency range results in sensitivity increases over another frequency range.

In this section two approaches to partial LTR applicable to NMP plants are described.

Trial Approach

One approach to applying loop recovery techniques to NMP plants is to proceed tentatively as if the plant were minimum phase, just increasing noise variances (fictitious noise) until robustness is maximized in some sense, never expecting full recovery of state feedback open-loop properties, nor expecting continual improvement in robustness as the noise variance increases [41].

Factored Plant Model Approach

This method uses an all-pass/minimum-phase (i.e., inner/outer) factored form for the NMP plant [42]. Consider the linear model (7.1) with transfer function $W(z)$. Consider also an inner/outer factorization of $W(z)$ as

$$W(z) = W_i(z)W_o(z) \tag{7.5}$$

Here $W_i(z)$ is asymptotically stable, and "inner" or equivalently "all-pass"; and $W_o(z)$ is "outer" or "minimum-phase". The all-pass W_i is characterized by a state-space representation with state vector denoted by x_i and associated system matrices $\{F_i, G_i, H_i, J_i\}$. Similarly, the minimum-phase system W_o is characterized by state vector x_o and state matrices $\{F_o, G_o, H_o\}$. This factored model is depicted in Figure 7.2. Not shown in Figure 7.2 are the measurement and process noises, which are included in equation (7.1). With this factored plant model, the state vector x of (7.1) is partition as $[x_o^T\, x_i^T]$; and F, G, M, H of (7.1) assume the following partitioned forms

$$F = \begin{bmatrix} F_o & G_oH_i \\ 0 & F_i \end{bmatrix}, \quad G = \begin{bmatrix} G_oJ_i \\ G_i \end{bmatrix}, \quad M = \begin{bmatrix} G_o & G_oJ_i \\ 0 & G_i \end{bmatrix}, \quad H = [H_o \quad 0] \tag{7.6}$$

The implication of the above partitioned M is that the process noise v also is partitioned,

$$v^T = [v_f^T\, v_p^T] \tag{7.7}$$

where v_p is the process noise added to the plant input, and v_f is a fictitious process noise added to the input of the minimum-phase factor W_o. In the design, v_f can be viewed as representing effects of plant uncertainty in the passband of W_o, and is used to achieve partial loop recovery properties. The noises are zero-mean Gaussian with covariances matrices $R_p > 0$ for w, $Q_p = q_pI \geq 0$ for v_p and $Q_f = q_fI > 0$ for v_f.

Applying Kalman filter theory [41] to the stochastic NMP plant model (3) yields the time invariant estimator

$$\hat{x}(k+1) = F\hat{x}(k) + Gu(k) + K_e(y - H\hat{x}(k)) \tag{7.8}$$

where K_e is the estimator gain (steady-state solution to the Kalman filter equations [41]). In (7.8) use is made of the partitioned forms in (7.6) along with

$$\hat{x}^T = [\hat{x}_o^T \hat{x}_i^T], \quad K_e^T = [K_{eo}^T K_{ei}^T], \quad Q = M\begin{bmatrix} q_f I & 0 \\ 0 & q_p I \end{bmatrix} M^T, \quad R = R_p \tag{7.9}$$

For $Q_p = q_p I$, the partitioned Riccati equation solution P of the Kalman filter has partial parts $P_{12} = P_{21}^T = 0$, and also both $K_{ei} = 0$ and $\lim_{q_f \to \infty} \hat{x}_o = x_o$.

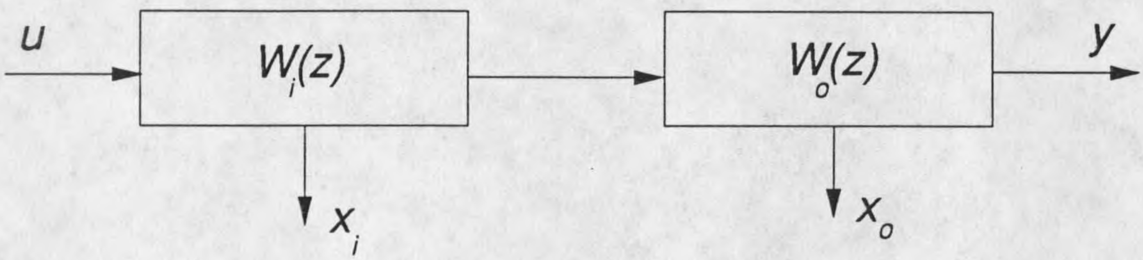

Figure 7.2. Stochastic factored NMP model for plant.

Thus the estimator gain is constructed in terms of only the outer factor parameters $\{F_o, G_o, H_o\}$ and the covariances $R, q_p I$ and $q_f I$. By increasing fictitious process noise v_f the open-loop properties of certain partial state feedback designs are recovered in the state estimation feedback controller design.

Test System Description

The power system example described here consists of an LQG/LTR controller applied to a static var compensator (SVC) in a low-order model power system (Figure 5.2).

The SVC is placed at the bus SOCAL. The primary purpose of the SVC is local bus voltage control. Interarea damping control is added as a supplementary control loop which utilizes line power between NOCAL and SOCAL as the input signal. This signal was selected as the controller input because it has good controllability and observability characteristics for several interarea modes.

Simulation Results

Several different tests are described in this section. For the cases that involve added system noise, an approximation of Gaussian white noise is applied to change the load at bus NOCAL, and this produces colored noise in system signals.

Comparison of Two LQG/LTR Approaches for NMP Power Systems

In this subsection, the two approaches to partial loop transfer recovery are compared. Regarding the *trial approach*, Figures 7.3 and 7.4 show the effect of increasing fictitious process noise on the sensitivity of the state feedback (SF) model versus sensitivity of the state estimate feedback (SEF) model at the input of the plant model. The figures illustrate sensitivity reduction achieved in two frequency ranges (0.0 to 0.4 Hz and 0.6 to 1.3 Hz) results in sensitivity increase over other frequency ranges (0.4 to 0.6 Hz, and 1.3 to 2.8 Hz). The frequencies of interarea modes are between 0.3 Hz and 0.9 Hz. The peak in Figures. 7.3 and 7.4 around 0.5 Hz shows the effect of plant NMP zeros. Increasing fictitious process noise (to some extent) has a good effect on the robustness of the controller when applied to the power system.

Regarding the factored plant model approach to achieve partial loop recovery, Figures. 7.5 and 7.6 illustrate the effect of increasing fictitious process noise on the sensitivity of the outer state feedback (OSF) model versus the sensitivity of the state estimate feedback (SEF) model at the input of the plant model. We can see how loop recovery is achieved in the frequency range of interest. Increasing the fictitious process noise in the LQG/LTR design for this case results in substantial improvement in the robustness of the controller.

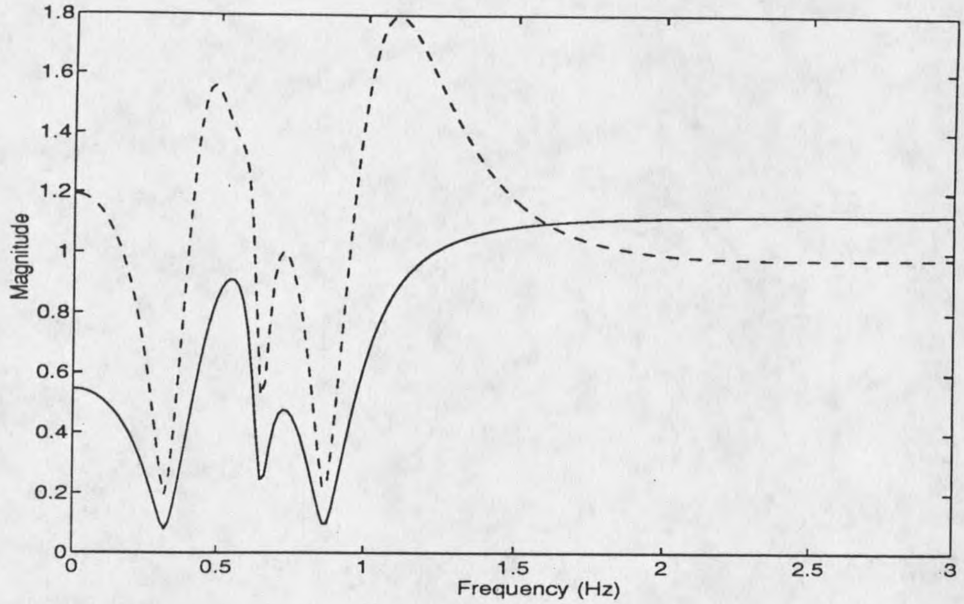

Figure 7.3. Sensitivity of SF model (solid line) versus sensitivity of SEF model (dashed line) with low fictitious process noise in estimator design.

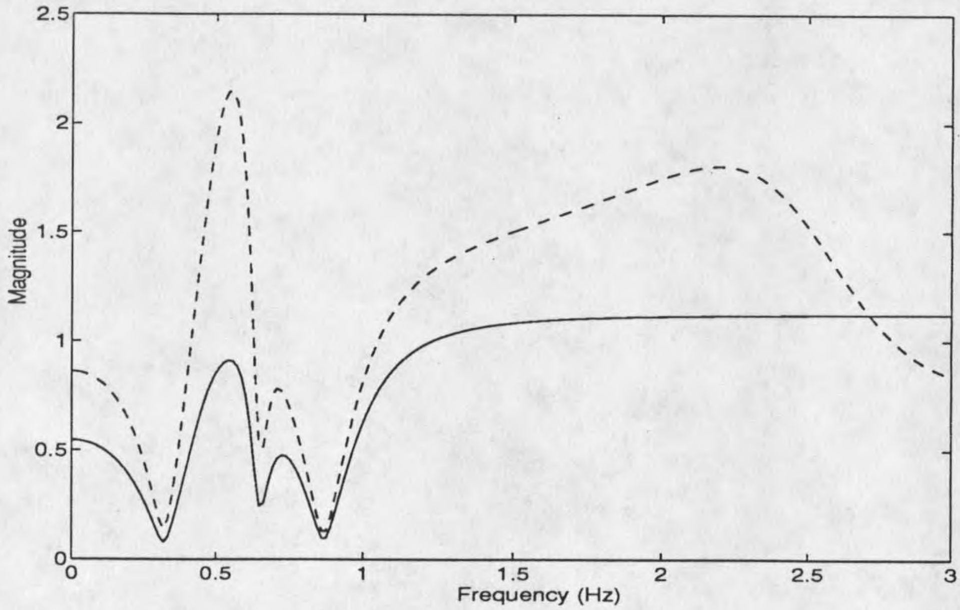

Figure 7.4. Sensitivity of SF model (solid line) versus sensitivity of SEF model (dashed line) with high fictitious process noise in estimator design.

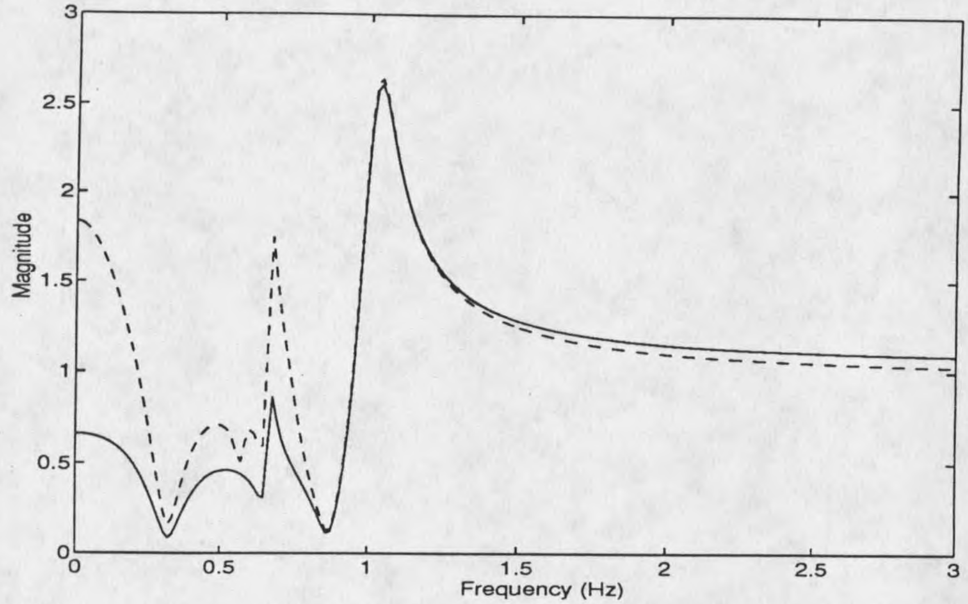

Figure 7.5. Sensitivity of OSF model (solid line) versus sensitivity of SEF model (dotted line) with low fictitious noise in estimator design.

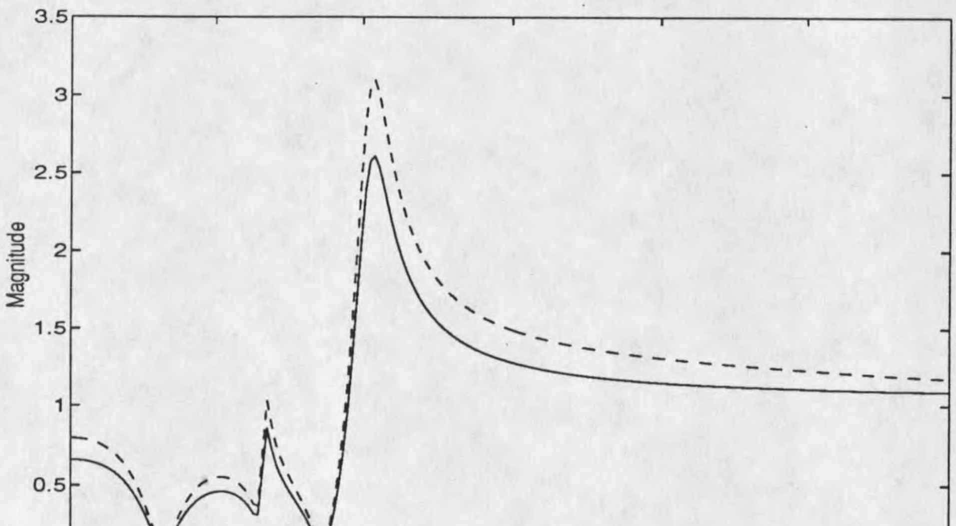

Figure 7.6. Sensitivity of OSF model (solid line) versus sensitivity of SEF model (dotted line) with high fictitious noise in estimator design.

Effects of Added Power System Noise

In this subsection, a high level of noise is added to the power system by random changes in the load at bus NOCAL. A combined three-phase fault and reduction in generation is also applied to the system (at NORTH). When no SVC damping control is applied, the fault and noise are sufficient to cause the system to go unstable after approximately 20 seconds. Figures 7.7 and 7.8 show the relative phase angle (between SOUTH and CANAD) responses that result when the controllers of the *inner/outer factorization designs* method are included in the system; the controller designed with low fictitious process noise exhibits an unstable response after 30 seconds (Figure 7.7), whereas that based on high fictitious process noise results in a stable response (Figure 7.8). Similar results are obtained when the controllers of the *trial design* method are included in the system. However, by increasing the level of noise added to the power system the controller of the *inner/outer factorization design* method still remained stable but the controller of the *trial design* method went unstable (Figures 7.9 and 7.10).

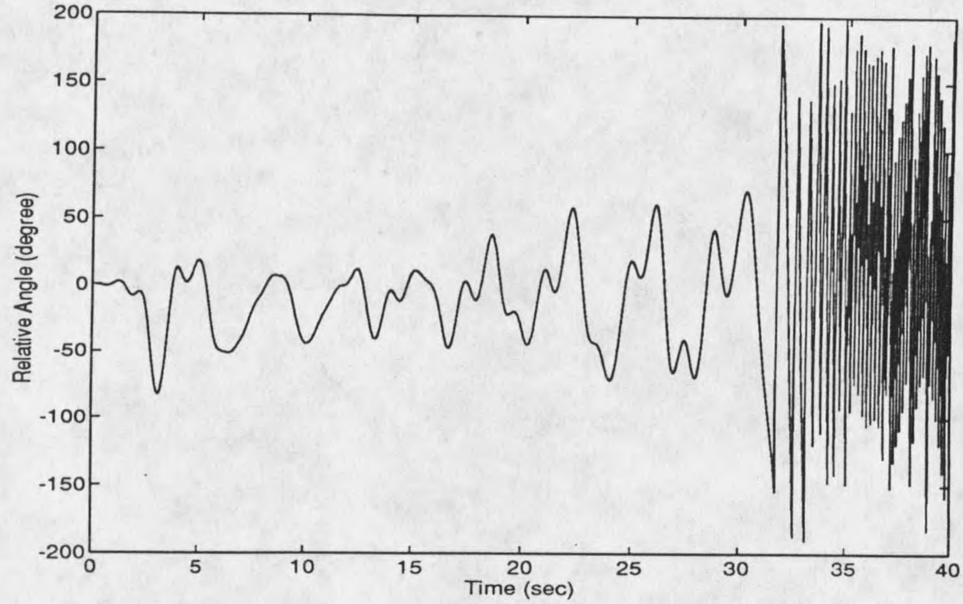

Figure 7.7. Relative phase angle with added noise with control(inner/outer factorization design with low fictitious process noise in estimator design).

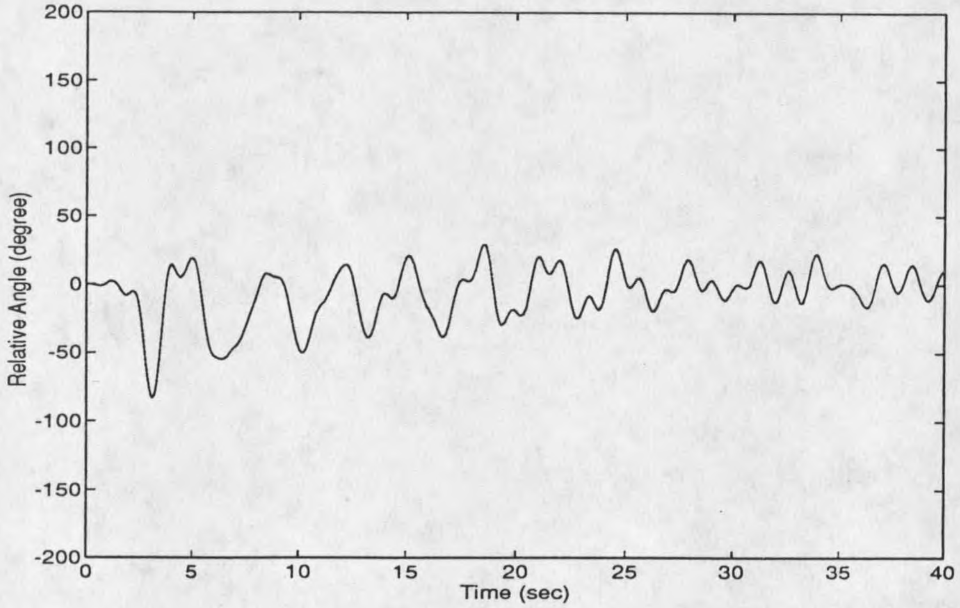

Figure 7.8. Relative phase angle with added noise with control (inner/outer factorization design with high fictitious process noise in estimator design).

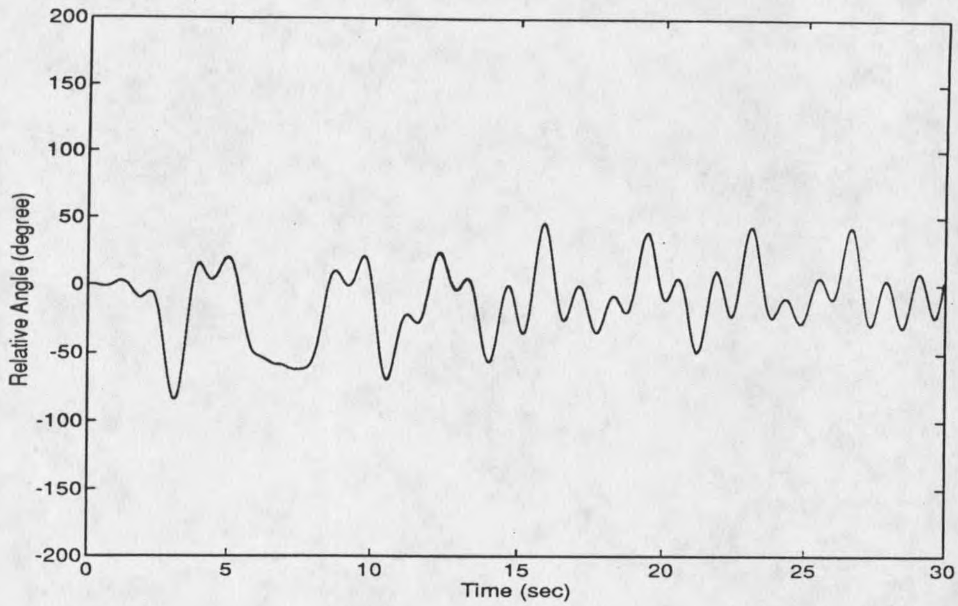

Figure 7.9. Relative phase angle with increased noise level (inner/outer factorization design with high fictitious process noise in estimator design).

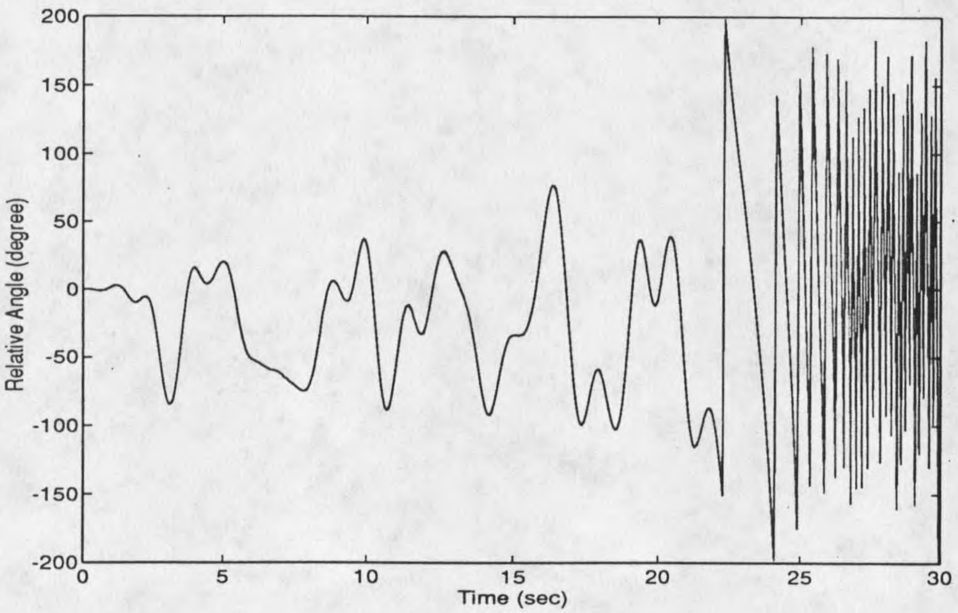

Figure 7.10. Relative phase angle with increased noise level (trial design with high fictitious process noise in estimator design).

Discussion of Results

Partial LQG/LTR controller design for NMP plants is addressed in this chapter. Controllers are designed to dampen oscillations in a low-order nonlinear power system model which has dynamic characteristics similar to the western North American power system. The control is applied to a static var compensator (SVC) in the system. The plant model associated with the controller is nonminimum phase (NMP) and stable. Two methods of partial loop transfer recovery are described and compared. The robustness of the resulting controllers is examined relative to the amount of loop transfer recovery that is possible for the NMP plant model. Nonlinear power system simulations are used to illustrate the relative effectiveness of the controllers under a variety of conditions, including three-phase faults, operating point changes, and load noise.

CHAPTER 8

CONCLUSION AND FUTURE WORK

In this thesis two approaches to robust controller design for SVC's are presented. Both of these approaches attempt to dampen low-frequency oscillations in a noisy multimachine power system. The simulations illustrate the effectiveness of the controllers in a realistic power system environment.

Chapter 2 describes the development and evaluation of a low-order power system transient stability model which has dynamic characteristics similar to those exhibited by a full-order model. The procedure for this model development is conceptually simple and yet the resulting system is quite adequate for developing and testing controllers for interarea damping. The model presented is a 46 bus system which is designed to have dynamic characteristics resembling those of a 2000 bus model of the western North American power system. The objective of the low-order modeling effort is to create a realistic environment for testing dynamic control strategies. Controller designs which appear promising on the test model can then be evaluated more thoroughly on the high-order system.

A possible extension of this model would include intermediate levels of modeling between the 46 bus system and the 2000 bus system. Representing one or more areas by detailed models and interconnecting them to other areas, ones modeled with less detail, will result in stability models which are suitable for studies where both local and interarea modes are important. Because detailed modeling is lacking in each of the areas of the low-order model, it may not reproduce the local dynamic environment with sufficient accuracy. However, control strategies applied to PSS units, SVC's, HVDC converters, or other FACTS devices in this test model will find a control environment that is more similar to an actual large power system model than many small or low-order models may provide.

Robust Adaptive Control

In Chapter 3 recursive extended least squares is incorporated in a self-tuning adaptive controller which uses an enhanced linear-quadratic control algorithm. Simulations show how effectively the controller performs on a lightly damped mechanical system in the presence of a significant amount of process noise. In Chapter 4 the self-tuning adaptive controller of Chapter 3 is used to dampen low-frequency oscillations in simulations of large-scale power systems. Cases with and without noise included were used to illustrate the robustness of the controller to system noise; in cases with substantial system noise, the RELS identifier generally produced better results than those obtained with the RLS identifier.

It is shown that in the presence of process noise, self-tuning adaptive controllers need to use continued probing to maintain a reasonable model of the plant. The combined effect of the process noise, continued probing, and adaptive control is generally to decrease the oscillatory amplitudes in system signals.

The key is to select a probing signal level that is neither too small (having insignificant impact compared to the noise) nor too large (driving system responses to amplitudes well beyond those produced by the noise alone).

While the results displayed in Chapter 4 are encouraging, there are uncertainties associated with the application of adaptive control in large power systems. Both the power system and the control depend on many parameters. For the control, adjustments can be made in many features, including performance weights, identifier properties, probing level, SVC parameters, and the types of signals used for feedback. The power system structure and noise level can vary dramatically. Some changes can cause the adaptively controlled SVC to enter a limit cycle producing large oscillations throughout the power system. A significant amount of trial and error is needed to determine appropriate parameters for successful operation of the adaptive controller in this environment. Also, in some instances, disturbances can result in oscillations which are not controllable by a particular SVC. In the cases illustrated in Chapter 4, however, the controller was effective in damping the dominant 0.3 Hz mode.

When disturbances occur in large power systems, many control devices can be active, and the response exhibited by a given network signal is the result both of disturbances and of the control actions of multiple controllers. It is important, therefore, that the identifier of each adaptive controller in such a system be monitored in such a way that it does not produce grossly inaccurate transfer function models. The only way to assure this is to create supervisory software for each identifier to dictate when input-output data is to be used by the identifier. In any case where a given adaptive controller is used as a PSS unit in an exciter loop of a generator, the dynamic response of that generator is dominated by the PSS unit, and recursive identification need only be interrupted during those times when severe faults occur close to the given machine. However, with an adaptive SVC damping device that is associated with multiple machines, the network signals upon which the control is based can contain significant components resulting both from remote faults and from interarea modes of oscillation that may not be controllable by the given SVC even though they are observable. A recursive identifier that uses such system response data may generate a grossly inaccurate system model, and this may lead to destabilizing adaptive control. Thus, a major contribution to the field of adaptive damping control using SVC's would be the development of reliable supervisory criteria to determine which real time data to use in model identification.

System Identification and Robust Nonadaptive Control

In Chapter 5 an introduction to concepts and applications of transfer function identification in power systems is presented. The chapter begins with a brief introduction to transfer function identification methods using least-squares approaches and then discusses applications which include: SVC's, model validation applications and software validation. A comparison is also made between eigenvalues obtained from transfer function identification and small signal analysis. The results between these entirely different approaches show good agreement which indicates that both methods can be valid ways of obtaining key parameters of power system dynamics.

In Chapters 6 and 7 a new nonadaptive control technique has been presented to dampen low-frequency oscillations in power systems.

An iterative closed-loop identification method is applied to obtain linear models of the power system for use in controller design. Although the control is applied to a static var compensator (SVC) in the system, the design process presented can in theory be readily applied to many other power system devices. Two different probing levels are used for model identification to demonstrate both the underlying nonlinear nature of the power system and the effectiveness of using the closed-loop iterative identification method in the case of large probing inputs. The identified plant models contain nonminimum phase zeros, and controllers are designed using linear quadratic Gaussian theory with partial loop transfer recovery (LQG/LTR).

Two methods of partial loop transfer recovery are described and compared. The robustness of the resulting controllers is examined relative to the amount of loop transfer recovery that is possible for the NMP plant model. Nonlinear power system simulations are used to illustrate the relative effectiveness of the controllers under a variety of conditions, including three-phase faults, operating point changes, and load noise.

The relative success of the nonadaptive controller by using the iterative method and the LQG/LTR control strategy indicates that more work should be done in this area. Robust methods of identification that utilize models which include modeled uncertainty should be developed for identifing power system models. The identifier should also consider the effects of noise in the system.

Another possible suggestion is to use a combined iterative control system design which couples the separate stages of frequency weighted least-squares model identification with controller design based on frequency weighted LQG methods [35,36].

REFERENCES CITED

[1] Electric Power Research Institute, "Extended Transient Midterm Stability Package: Final Report," EPRI TR-102004 Project 1208, Final Report, April 1993.

[2] J. R. Smith, F. Fatehi and C. S. Woods, "A Manual for the HOOK Interface to ETMSP," Report on Task 1 BPA Project, Electrical Engineering Dept., Montana State University, Bozeman, 1994.

[3] J. F. Hauer, "A Perspective on the Need for Robust Damping Controls in the Western U. S. Power System," *IEEE Control Systems Magazine*.

[4] J. R. Smith, "Robust Var Unit Control Strategies for Damping of Power System Oscillations," Ph.D. Thesis, Montana State University, Bozeman, Montana, July 1988.

[5] Laszlo Gyugyi, "Fundamentals of Thyristor-Controlled Static Var Compensators in Electric Power System Applications," *IEEE Winter Power Meeting, Publication 87TH0187-5-PWR*, 1987.

[6] E. Lerch, D. Povh and L. Xu, "Advanced SVC Control for Damping Power System Oscillations," *IEEE Trans. on Power Systems*, Vol. 6, No. 2, pp. 524-535, May 1991.

[7] K. R. Padiyar and R. K. Varma, "Damping Torque Analysis of Static VAR System Controllers," *IEEE Trans. on Power Systems*, Vol. 6, No. 2, pp. 458-465, May. 1991.

[8] E. V. Larsen, D. H. Baker, A. F. Imece, L. G. Lajoie and G. Scott, "Basics Aspects of Applying SVC's to Series-Compensated AC Transmission Lines," *IEEE Trans. on Power Delivery*, Vol. 5, No. 3, pp. 1466-1472, July 1990.

[9] P. K. Dash, A. M. Sharf and E. F. Hill, "An Adaptive Stabilizer for Thyristor Controlled Static Var Compensators for Power Systems," *IEEE Trans. on Power Systems*, Vol. 4, No. 2, pp. 403-410, May 1989.

[10] Y. Y. Hsu, C. S. Liu, C. J. Lin and C. T. Huang "Application of Power System Stabilizers and Static Var Compensators on a Longitudinal Power System," *IEEE Trans. on Power Systems*, Vol. 3, No. 4, pp. 1464-1469, Nov. 1988.

[11] M. M. Gupta (Editor), *Adaptive Methods for Control System Design*, IEEE Press Selected Reprint Series, IEEE Press, New York, 1986.

[12] K. S. Narendra, R. Ortega and P. Dorato (Editors), *Advances in Adaptive Control*, A Selected Reprint Volume, IEEE Press, New York, 1991.

[13] D. A. Pierre, "A Perspective on Adaptive Control of Power Systems," *IEEE Transactions on Power Systems*, Vol. PWRS-2, No. 2, pp. 387-396, May 1987.

[14] D. A. Pierre, "Enhanced LQ Controllers and Adaptive Control," *Proceedings*, IEEE Conference on Decision and Control, pp. 782-785, Dec. 1988.

[15] D. A. Pierre, "Properties of a Discrete-time Enhanced Linear-Quadratic Controller," *Automatica*, Vol. 27, No. 6, pp. 1029-1034, 1991.

[16] D. A. Pierre and D. J. Trudnowski, "Two Formulations of Exponentially Weighted Enhanced LQ Controllers," *IEEE Trans. on Automatic Control*, Vol. 38, No. 7, July 1993.

[17] C. Samson, "An Adaptive LQ Controller for Nonminimum-Phase Systems," *International J. Control*, Vol. 35, No. 1, pp. 1-28, 1982.

[18] J. R. Smith, D. A. Pierre, D. A. Rudberg, I. Sadighi, A. P. Johnson and J. F. Hauer, "An Enhanced LQ Adaptive VAR Unit Controller for Power System Damping," *IEEE Trans. on Power Systems*, Vol. 4, pp. 443-451, May 1989.

[19] J. R. Smith, D. A. Pierre, D. A. Rudberg, I. Sadighi, M. H. Nehrir and J. F. Hauer, "A Supplementary Adaptive VAR Unit Controller for Power System Damping," *IEEE Trans. on Power Systems*, Vol. 4, pp. 1017-1023, Aug. 1989.

[20] D. J. Trudnowski, "Decentralized Adaptive Control and System Identification, with Applications to Power Systems," Ph.D. Thesis, Montana State University, Bozeman, Montana, March 1991.

[21] D. J. Trudnowski and D. A. Pierre, "A Decentralized Indirect Adaptive Control Scheme Applicable to Overlapping Interconnected Systems," *International Journal of Control*, Vol. 55, No. 2, pp. 343-360, 1992.

[22] D. A. Pierre, I. Sadighi and D. J. Trudnowski, "System Identification and Decentralized Adaptive Control with Applications to Power Systems," EPRI TR-101097, Vol. 1 of Final Report on EPRI Project RP-2665-01, Sept. 1992.

[23] D. J. Trudnowski, D. A. Pierre, J. R. Smith and R. Adapa, "Coordination of Multiple Adaptive PSS Units Using a Decentralized Control Scheme," *IEEE Trans. of Power Systems*, Vol. 7, No. 1, pp. 294-300, Feb. 1992.

[24] F. Fatehi and D. A. Pierre, "The Robustness of an Advanced Adaptive Controller Against System Noise," *Proceedings*, IEEE Conference on Decision and Control, pp. 587-593, Dec. 1993.

[25] F. Fatehi, D. A. Pierre and J. R. Smith, "Effect of Noise on the Adaptive Damping of Low-Frequency Oscillations in Power Systems," *Proceedings*, IEEE Conference on Decision and Control, pp. 2358-2363, Dec. 1993.

[26] G.S. Hope, S.T. Nichols and J. Carr, "Measurement of Transfer Functions of Power System Components Under Operating Conditions," *IEEE Trans. PAS*, Vol. PAS-96, No. 6, pp. 1798-1808, November/December 1977.

[27] D. J. Trudnowski, J. R. Smith, T. A. Short and D. A. Pierre, "An Application of Prony Methods in PSS Design for Multimachine Systems," *IEEE Trans. on Power Systems*, Vol. 6, No. 1, pp. 118-126, Feb. 1991.

[28] I. Kamwa, R. Grondin, E.J. Dickinson and S. Fortin, "A Minimal Realizatin Approach to Modal Analysis and Reduced-Order Modeling for Power System Signal Responses," *IEEE Transactions on Power Systems*, Vol. 8, No. 3, pp. 1020-1029, Aug. 1993.

[29] J. R. Smith, F. Fatehi, C. S. Woods, J. R. Hauer and D. J. Trudnowski, "Transfer Function Identification in Power System Applications," *IEEE Trans. on Power Systems*, Vol. T-PWRS-8, No. 3, pp. 1282-1290, Aug. 1993.

[30] D. A. Pierre, J. R. Smith, D. J. Trudnowski and J. W. Pierre, "General Formulation of a Prony Based Method for Simultaneous Identification of Transfer Functions and Initial Conditions," *Proceedings*, IEEE Conference on Decision and Control, pp. 1686-1691, Dec. 1992.

[31] J. R. Smith, C. S. Woods, F. Fatehi, G. L. Keenan and J. F. Hauer, "A Low Order Power System Model With Dynamic Characteristics of the Western North American System," *Proceedings*, North American Power Symposium, Sept. 1994.

[32] Ruud J. P. Schrama "Accurate Identification for Control: The Necessity of an Iterative Scheme" *IEEE Trans. on Automatic Control*, Vol. 37, No. 7, pp. 991-994, July 1992.

[33] D. S. Bayard, Y. Yam and E. Mettler "A Criterion for Joint Optimization of Identification and Robust Control" *IEEE Trans. on Automatic Control*, Vol. 37, No. 7, pp. 986-991, July 1992.

[34] F. R. Hansen, G. F. Franklin and R. Kosut, "Closed-Loop Identification via Fractional Representation: Experiment Design," *Proceedings*, American Control Conference, Pittsburgh, PA, pp. 1422-1427, 1989.

[35] Z. Zang, R. R. Bitmead and Gevers, "Iterative Model Refinement and Control Robustness Enhancement," *Proceedings*, IEEE Conference on Decision and Control, pp. 279-284, Dec. 1991.

[36] Z. Zang, R. R. Bitmead and Gevers, "Disturbance Rejection: Online Refinement of Controllers by Closed Loop Modelling," *Proceedings*, American Control Conference, pp. 2829-2833, 1992.

[37] A. G. Partanen and R. R. Bitmead, "Two Stage Iterative Identification-/Controller Design and Direct Experimental Controller Refinement," *Proceedings*, IEEE Conference on Decision and Control, pp. 2833-2838, Dec. 1993.

[38] R. R. Bitmead, M. Gevers and V. Wertz, *Adaptive Optimal Control- The Thinking Man's GPC,* Prentice Hall International, Series in System and Control Engineering, 1990.

[39] J. C. Doyle and G. Stein "Robustness with Observers" *IEEE Trans. on Automatic Control*, Vol. AC-24, No. 4, pp. 607-611, Aug. 1979.

[40] J. M. Maciejowski, *Multivariable Feedback Design*, Addison-Wesley Publishing Company, 1989.

[41] B. D. O. Anderson and J. B. Moore, *Optimal Control*, Prentice Hall, Englewood Cliffs, New Jersey 1990.

[42] J. B. Moore and Lige Xia, "Loop Recovery and Robust State Estimate Feedback Design," *IEEE Trans. on Automatic Control*, Vol. AC-32, No. 6, pp. 512-517, July 1987.

[43] Z. Zhang and J. S. Freudenberg, "Loop Transfer Recovery with Non-Minimum Phase Zeros," *Proceedings*, IEEE Conference on Decision and Control, pp. 956-957, Dec. 1987.

[44] J. R. Smith, F. Fatehi and D. A. Pierre, "Closed-Loop Identification and Tuning for Damping of Interarea Modes," *Proceedings*, IEEE Conference on Decision and Control, Dec. 1994.

[45] F. Fatehi, J. R. Smith and D. A. Pierre, "Robust Power System Controller Design Based on Measured Models,", IEEE Summer Power Meeting, July 1995, pending.

[46] P.A. Emmanuel, "Damping of Low Frequency Oscillations in AC/DC Power Systems Using HVDC Converter Control," Thesis, Montana State University, Bozeman, Montana, Oct. 1991.

[47] J.F. Hauer, "Power System Identification by Fitting Structured Models to Measured Frequency Response," *IEEE Transactions on Power Apparatus and Systems*, Vol PAS-101, No. 4, pp. 915-922, April 1982.

[48] J. F. Hauer, "The Use of Prony Analysis to Determine Modal Content and Equivalent Models for Measured Power System Response," IEEE publication No. 90TH0292-3-PWR entitled *Eigenanalysis and Frequency Domain Methods for System Dynamic Performance*, pp. 105-115, 1989.

[49] Y. Mansour, "Application of Eigenanalysis to the Western North American Power System," *Eigenanalysis and Frequency Domain Methods for System Dynamic Performance*, IEEE publication number 90TH0292-3-PWR, pp. 97-104, 1989.

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

[51] J. Arrillaga, C. D. Arnold and B. J. Harker, *Computer Modelling of Electrical Power Systems*, John Wiley and Sons, Inc., New York, 1983

[52] P. Kundur, G.J. Rogers, D.Y. Wong, L. Wang and M.G. Lauby, "A Comprehensive Computer Program Package for Small Signal Stability Analysis of Power Systems," *IEEE Transactions on Power Systems*, Vol. 5, No. 4, pp. 1076-1083, Nov. 1990.

[53] Small Signal Stability Program:Version 3.0, Ontario Hydro and EPRI project 2447-1, Dec. 1991.

[54] J. F. Hauer, C. J. Demeure and L. L. Scharf, "Initial Results in Prony Analysis of Power System Response Signals," *IEEE Transactions on Power Systems*, Vol. 5, No. 1, pp. 80-89, Feb. 1990.

[55] M. Klein, G.J. Rogers, S. Moorty and P. Kundur, "Analytical Investigation of Factors Influencing Power System Stabilizers Performance," paper # 92 WM 016-6 EC, presented at the IEEE Power Engineering Society Winter Meeting, New York, January 26-30, 1992.

[56] I. Sadighi, "System Identification Methods for Adaptive Control," Ph.D. Thesis, Montana State University, Bozeman, Montana, Oct. 1989.

[57] F. Fatehi and D. A. Pierre, "Robustness of an Adaptive Controller Against System Noise," Report on Task 6 BPA Project 1991-1993, Oct. 1992.

[58] B. J. Kuo, *Automatic Control Systems*, 6th Edition, Prentice Hall, New Jersey, 1991.

[59] N. Martins, "Efficient Eigenvalue and Frequency Response Methods Applied to Power System Small Signal Stability Studies," *IEEE Trans. on Power Systems*, Vol. PWRS-1, pp. 217-226, Feb. 1986.

[60] N. Martins, H. Pinto and L. Lima, "Efficient Methods for Finding Transfer Function Zeroes of Power Systems," 1991 *PICA Conference*, Baltimore, Maryland, May 1991.

[61] D. A. Pierre, D. J. Trudnowski and J. F. Hauer, "Identifying Reduced Order Models for Large Nonlinear Systems with Arbitrary Initial Conditions and Multiple Outputs using Prony Signal Analysis," *Proceedings*, American Control Conference, pp. 149-154, May 1990.

[62] M. Bounou, S. Lefebvre and R.P. Malhame, "A Spectral Algorithm for Extracting Power System Modes from Time Recordings," *IEEE Trans. on Power Systems*, Vol. 7, No.2, pp. 665-672, May 1992.

[63] P. B. Silverstein, O. S. Sands and F. D. Garber, "Radar Target Classification and Interpretation by Means of Structural Descriptions of Backscatter Signals," *IEEE Aerospace and Electronic Systems Magazine*, Vol 6, No. 5, May 1991.

[64] R. Kumaresan, D. W. Tufts and L. L. Scharf, " Prony Method for Noisy Data: Choosing the Signal Components and Selecting the Order in Exponential Signal Models." *Proceedings of the IEEE*, Vol. 72, No. 2, pp. 230-233, Feb. 1984.

[65] G. H. Hostetter, *Digital Control System Design*, Holt Rinehart and Winston Inc., New York, 1988.

[66] D. W. Tufts and R. Kumaresan, " Singular Value Decomposition and Improved Frequency Estimation Using Linear Prediction," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol ASSP-30, No. 4, pp. 671-675, Aug. 1982.

[67] L.L. Scharf, *Statistical Signal Processing: Detection, Estimation, and Time Series Analysis*, Addison-Wesley, New York, 1991.

[68] T. McWhorter and L.L. Scharf, "Cramer-Rao Bounds for Modal Analysis," Tech. Report DSP-S04, Electrical Engineerig Department, University of Colorado, Boulder, 1991

APPENDICES

APPENDIX A

A Manual for the HOOK Interface to ETMSP

A MANUAL FOR THE HOOK INTERFACE TO ETMSP

Introduction

The basic function of the subroutine HOOK is to allow access to any system signal within the ETMSP program. It also allows modification of controller reference signals and key device variables. In this way HOOK allows the user to implement any type of controller which can be coded in FORTRAN. It also allows printing of any of these signals in a format of the users choosing. This feature supplements the output capability available in ITOAP.

HOOK was written with the intention of allowing an engineer using ETMSP to experiment with nonconventional and digital controllers. The idea was that the program would be open-ended enough so that the user would be minimally constrained. HOOK executes an interface to ETMSP, providing a path to retrieve signals from, and to input signals to ETMSP at each time step in the integration. It also allows access to signals during the run for the purposes of signal analysis, controls or special outputs. HOOK is called from the subroutine SIMULA at each time step. HOOK was designed to be able to be used without detailed knowledge of how ETMSP stores information. It should be noted, however, that ETMSP is a large, complicated program and for the very reason that HOOK allows the user to access global information (through common block allocation) it should be used with caution.

To use the HOOK subroutine, it is nescessary to complete five user blocks in a HOOK template. This version of HOOK then must be linked with a modified version of SIMULA called SIMULAX and the rest of ETMSP

Program Outline

HOOK is divided to nine different parts, five of which are intended to be completed by the user. Parts three through six are executed only the first time that the routine HOOK is called, at TS = 0. The parts are as follows:

1) The common blocks are included and the real, character and integer variables which are used in the subroutine are defined.
2) (User block 1) The user defines new variables which will be used in the routine using the REAL, INTEGER, and DIMENSION fortran statements.
3) The output data file "HK.DAT" is created.
4) (User block 2) The user assigns values for the integer variables POUT, NT, and the character array BSNM(I) for each I from 1 to NT. POUT is the number of iterations between writes to HK.DAT. NT is the number of buses being monitored and the BSNM(I)'s are the names of the buses. (See Section III)
5) The subroutine gets bus numbers in internal order, assigning them to the integer array PT(I) where the I's are in the same order as for BSNM(I).
6) (User block 3) The user retrieves the indices of any relevant states as well as any necessary dynamic data and/or initial conditions of the system states. (See Section III)
7) HOOK checks for repeated calls at the same time because of discontinuities.

8) (User block 4) The user retrieves the states, which are accessed with the indices found in part six, and uses these variables with any relevant system data to take control action.

9) (User block 5) The user writes any desired states or variables to the data file "HK.DAT" using the fortran statements WRITE and FORMAT.

ACCESSING ETMSP VARIABLES

The process of gaining access to variables within ETMSP consists basically of knowing three pieces of information. The first is the relevant machine or bus numbering, the second is the name of the array which contains the variable needed, and the third is the index of the array corresponding to a particular machine, bus, or block in a block diagram. ETMSP has the entries in arrays in different ordering depending on where the program is in the solution cycle. When the program first starts the buses are assigned a number. This is the loadflow ordering of the buses. ETMSP has a routine called CHKBUS; if the user sends CHKBUS the bus name, it returns the bus number in loadflow ordering. HOOK uses the routine CHKBUS to obtain the loadflow number of buses from the bus name and stores them in BNUM(I). In ETMSP there are several arrays which convert the loadflow bus number to another index number to access a variable associated with that bus. As an example, the bus index number (in internal order), PT(I), can be obtained using NNEW(BNUM(I)). PT(I) can then be used in the array MSTATE(1,PT(I)) to obtain the index PTS=MSTATE(1,PT(I)). This index PTS is then used in the array XSTATE(PTS+2) to obtain the angle of the machine corresponding to the bus PT(I).

So the process for gaining access to a variable is:

1) Get bus (or generator) number in the appropriate order.
2) Use an index variable to get the appropriate index for the desired variable
3) Use this index variable (and some constant) in the appropriate array to get at the desired variable.

Of the three steps outlined above, the second two are very straight-forward. Due to the fact that ETMSP numbers the buses (and generators) in different ways throughout the program, the first step is the tricky one. We have worked with detailed generators and their associated controls. These can be accessed using PT(I), but only in the case that there is a single machine per bus. We have also worked with SVC's and are confident of the indexes necessary to access their states.

The following series of lists contain some of the state variables and dynamic data that can be accessed in ETMSP using HOOK. The lists are by no means exhaustive. We will continue to expand them as we become aware of new information. Notice that the variables are accessed using the index variables which are included at the beginning. These index variables are generally accessed using PT(I) which is generated by the array NNEW when it is given the argument BNUM, the bus number in loadflow ordering.

Detailed machine models:

As mentioned above, when accessing data associated with detailed machines, PT(I) can be used if one machine is connected per bus.

Variables containing indices

LOADFLOW BUS NUMBER	BNUM
BUS NUMBER(INTERNAL ORDER)	PT(I)
MACHINE Y-VARIABLE FIELD OFFSET	MYVAR(1,PT)
MACHINE STATE FIELD OFFSET	MSTATE(1,PT)
MACHINE DATA FIELD OFFSET	JCONTA(1,PT)
GOVERNOR Y-VARIABLE FIELD OFFSET	MYVAR(5,PT)
GOVERNOR STATE FIELD OFFSET	MSTATE(5,PT)
GOVERNOR DATA FIELD OFFSET	JCONTA(2,PT)
EXCITER Y-VARIABLE FIELD OFFSET	MYVAR(2,PT)
EXCITER STATE FIELD OFFSET	MSTATE(2,PT)
EXCITER DATA FIELD OFFSET	JCONTA(3,PT)

These indices are then used to access the data in the following manner:

Variable Name	Variable	Index
SPEED	XSTATE(* + 1)	* = MSTATE(1,PT)
ANGLE	XSTATE(* + 2)	* = MSTATE(1,PT)
MECHANICAL TORQUE	TIN(* + 1)	* = MYVAR(1,PT)
POWER ELECTRICAL(ACTIVE)	PDG(* + 2)	* = MYVAR(1,PT)
POWER ELECTRICAL(REACTIVE)	PQG(* + 3)	* = MYVAR(1,PT)
FIELD VOLTAGE	FED(* + 4)	* = MYVAR(1,PT)
FIELD CURRENT	CFD(* + 5)	* = MYVAR(1,PT)
GENERATOR VOLTAGE	VOLT(*)	* = PT
REAL CURRENT	CDG(1 , *)	* = PT
IMAGINARY CURRENT	CDG(2 , *)	* = PT
GENERATOR PARAMETERS	DDAT(* + i) i=1,2,...,49	* = JCONTA(1,PT)
Governor turbine		
GOV REFERENCE VALUE	XLDREF(* + 1)	* = MYVAR(5,PT)
GOVERNOR POWER OUTPUT	XSTATE(* + 2)	* = MSTATE(5,PT)
TURBINE POWER OUTPUT	TIN(* + 1)	* = MYVAR(1,PT)
TYPE OF HYDRO TURBINE	ICONTA(2 , *)	* = PT
GOV_TURBINE PARAMETERS	DDAT(* + i) i=1,2,...16	* = JCONTA(2,PT)
Exciter		
EXCITER REFERENCE VALUE	VREF(* + 2)	* = MYVAR(2,PT)
EXCITER OUTPUT	FED(* + 4)	* = MYVAR(2,PT)
EXCITER OUTPUT	XSTATE(* + 2)	* = MSTATE(2,PT)
EXCITER TYPE	ICONTA(4 , *)	* = PT
EXCITER PARAMETERS	DDAT(* + i) i=1,2,...,27	* = JCONTA(3,PT)

Classical machine models:

To access classical machine data the index NC is needed.

Variables Containing Indices:

CLASSICAL MACHINE IN INTERNAL ORDER	NC
MACHINE Y-VARIABLE FIELD OFFSET	MYVARB(1,NC)
MACHINE STATE FIELD OFFSET	MSTATB(1,NC)
MACHINE DATA FIELD OFFSET	JCONTB(1,NC)

Variable Name	Variable	Index
ANGLE	XSTATE(* + 2)	* = MSTATB(1,NC)
SPEED	XSTATE(* + 1)	* = MSTATB(1,NC)
MECHANICAL TORQUE	TIN(* + 1)	* = MYVARB(1,NC)
POWER ELECTRICAL(ACTIVE)	PDG(* + 2)	* = MYVARB(1,NC)
POWER ELECTRICAL(REACTIVE)	PDG(* + 3)	* = MYVARB(1,NC)
FIELD VOLTAGE	FED(* + 4)	* = MYVARB(1,NC)
GENERATOR PARAMETERS	DDAT(* + i) i=1,2,...,5	* = JCONTAB(1,NC)

User defined SVC (TYPE 50):

When accessing any SVC data, it is necessary to know the SVC number, N. ETMSP assigns these numbers sequentially starting with 1 in the order in which the SVC's are read in from the dynamic data file. The user defined SVC index variable and the user defined SVC state variables are included together in the table below.

Variable Name	Variable	Index
SVC Y-VARIABLE FIELD OFFSET		JVSVR(N)
REFERENCE INPUT VALUE	RVREF(* + 1)	* = JVSVR(N)
SVC BUS VOLTAGE	VOLT(*)	* = NROLD(N)
SVC SUSCEPTANCE	BSVC(* + 2)	* = JVSVR(N)
TCR SUSCEPTANCE	BTCR(* + 6)	* = JVSVR(N)
TCR CONTROL SIGNAL	STCR(* + 7)	* = JVSVR(N)

The array VALIN (listed below) contains feedback variables associated with user defined SVC's. Notice, however, that the SVC number N is not included in the indexing of this information. This is because the same array is repeatedly overwritten during each iteration of ETMSP. If an entry of VALIN is accessed in HOOK, the value will be for the last user defined SVC processed in the previous iteration of ETMSP. This will always be the user defined SVC with the highest N. If only a single user defined SVC is on the system then this is not a problem. For internally modeled SVC's, the array VALIN is not used. The elements of VALIN are listed below.

User Defined SVC Feedback Variables

SVC CURRENT(ISVC)	VALIN(1)	
TCR CURRENT(ITCR)	VALIN(2)	
SVC REACTIVE POWER(QSVC)	VALIN(3)	
TCR REACTIVE POWER(QTCR)	VALIN(4)	
SVC SUSCEPTANCE(BSVC)	VALIN(5)	
TCR SUSCEPTANCE(BTCR)	VALIN(6)	
SVC VOLTAGE(VSVC)	VALIN(7)	
ANGLE OF VSVC(ASVC)	VALIN(9)	
TCR SIGNAL(STCR)	VALIN(11)	
CONSTANT VALUE	VALIN(13)	
SIMULATION TIME(TIME)	VALIN(14)	
REFERENCE INPUT VALUE	VALIN(20)	
#REMOTE BUS VOLTAGE(VRxx)	VALIN(*)	* = 21 TO 30
ANGLE OF VRxx(ARxx)	VALIN(*)	* = 31 TO 40
REMOTE LINE CURRENT(IRxx)	VALIN(*)	* = 41 TO 50
REMOTE LINE POWER(PRxx)	VALIN(*)	* = 51 TO 60
REMOTE LINE REACTIVE POWER(QRxx)	VALIN(*)	* = 61 TO 70
REMOTE LINE POWER FACTOR	VALIN(*)	* = 71 TO 80

xx = 1 to 10 is the number of the remote bus or the remote line which is used in the dynamic data of the SVC.

SVC BLOCK OUTPUT	VALOUT(*)	* = NUMBER OF BLOCK

Variables for SVC:

The following variables apply to internally modeled SVC's. Notice that some of the variables are the same as for user defined SVC's.

Variable Name	Variable	Index
SVC Y-VARIABLE FIELD OFFSET		JVSVR(N)
REFERENCE INPUT VALUE	RVREF(* + 1)	* = JVSVR(N)
SVC SUSCEPTANCE	BSVC(* + 2)	* = JVSVR(N)
BUS VOLTAGE (for a bus which has an SVC)	VOLT(*)	* = NROLD(N)
SVC PARAMETERS (for a type 1 SVC)	DDAT(* + i) i=1,2,...,8	* = JSDAT(N)
SVC TYPE CODE	IDDAT(* + 9)	* = JSDAT(N)

EXAMPLES

In this section a number of example cases are presented. These example cases constitute the manner in which we have used HOOK. When using HOOK in ways other than those presented here the user should be very cautious. All examples are conducted using the 17 machine test case developed at Montana State University. A brief explanation of this model precedes the examples.

The 17 machine Western U.S. equivalent model

The western U.S. power system involves thousands of machines and hundreds of buses. For this reason, extensive CPU time is required for stability simulation studies involving a complete dynamical representation of the system. It is unfortunately the case that even using a complete dynamical model, it is not at all easy to represent the system in such a manner as to get simulation results which accurately reflect the physical behavior of the system. Given such a problem, it would seem unlikely that a reduced order model of the system would have any success at all in representing the system. The 17 machine equivalent model was therefore developed as a system with similar interarea modes to the actual system.

The physical phenomenon of all the machines in a geographical area of the system swinging, in phase with each other, against machines in another area of the system generates what are called interarea modes. The 17 machine model was developed as a test case for preliminary studies involving the identification and control of interarea modes. By representing whole areas of the physical system using a single machine with very large inertia, the interarea modal structure can be retained. Such a model would has local modes at all, and so, as can be seen in Figure 5.2, which illustrates the model, a number of smaller machines are included at each bus (area) so as to be able to match some of the dominate local modes. These smaller machines also have a small influence on the interarea modes. Simulation runs have shown that the modes in the 17 machine model are similar to the interarea modes found in simulations involving complete dynamical models of the system.

Another area in which the 17 machine model can be useful is in testing general sorts of control schemes and transfer fuction identification methods. It provides a sort of middle ground between a, far too simple, infinite bus type of approach and a tedious full system approach to researching power system behavior. In the examples that follow, the model is used in this latter manner.

Example 1

This example involves an EPRI type H hydro governor. In this case the base dynamic data file is modified by removing all the governors except the one at PACNW. Which is a type H hydro governor already. The block diagram for a type H hydro governor is shown in Figure 5-60 (ETMSP User's Manual). The data associated with each of the blocks in Figure 5-60 can be obtained either of two ways. The first is by carefully reading the dynamic data file and comparing it with the Governor Input Data page from the ETMSP documentation. The second method of obtaining the data is directly from the arrays in ETMSP where it is stored. The relevant arrays are listed for convenience in Table 1 below.

P_{max} = DDAT(INDX+1) = 13000
R = DDAT(INDX+2) = 0.05
T_G = DDAT(INDX+3) = 0.5
T_P = DDAT(INDX+4) = 0.04
T_D = DDAT(INDX+5) = 8.0
D_D = DDAT(INDX+6) = 0.4
$\frac{T_W}{2}$ = DDAT(INDX+9) = 1.0

Table 1

We wanted to use HOOK to provide a modification of the reference input to this governor so that the output signal could be analyzed using a numerical transfer function identification algorithm (prony). The user blocks necessary to modify the reference input P_oR to form a step of amplitude 0.2 from time TS = 0 to TS = 0.4 are presented below.

```
C       USER BLOCK 1
        REAL*4 ORIGVALUE, ANGL, SPSG, MECHPWR, GOPWO,                (1)
     &      ORIGMECPW, ORIGGOPW, DIFMECH, DIFGOPW                     (2)
        INTEGER IREFPT, IANG, IREFPT2, IMEC                          (3)
```

Be sure to define all variables that are being used in the user blocks. If, by an unlucky chance, one of the variables chosen is the same as a global variable used by ETMSP then a multiple definition error will show up during the linking.

```
C       USER BLOCK 2
        POUT = 5                                                     (1)
        NT = 1                                                       (2)
        BSNM(1) = 'PACNW_G120.0'                                     (3)
```

Here POUT = 5 will cause HOOK to write to the output file at TS = 0, TS = 5*(time step),10*(time step), etc. NT = 1 tells HOOK that one bus is to be monitored and the following line tells HOOK the name of this bus.

```
C       USER BLOCK 3
C       GENERATE FIELD OFFSET VARIABLES                              (1)
        IREFPT = MYVAR(5,PT(1))                                      (2)
        IREFPT2 = MSTATE(5,PT(1))                                    (3)
        IMEC = MYVAR(1,PT(1))                                        (4)
        IANG = MSTATE(1,PT(1))                                       (5)
C       GET GOV REFERENCE VALUE AT T=0                               (6)
        ORIGVALUE = XLDREF(IREFPT + 1)
```

```
C       GET MECHNICAL TORQUE AT T=0                                      (7)
        ORIGMECPW = TIN(IMEC + 1)                                        (8)
C       GET GOVERNOR OUTPUT POWER AT T=0                                 (9)
        ORIGGOPW = XSTATE(IREFPT2 + 2)                                   (10)
C       MAKE ANY INITIAL WRITE TO UNIT 21                                (11)
        WRITE(21,1000)                                                   (12)
1000    FORMAT('DATA WITH GOV REFERENCE MODIFIED ON PACNW_G120.0')       (13)
                                                                         (14)
```

In lines 2-4 offset variables are being defined. These index variables can then be used in user block 4 to access the states. Note that the index IREFPT2 will be used to access all governor states, IREFPT will be used to access all governor Y-variables, IMEC will be used to access machine Y-variables, and IANG will be used to access machine state variables.

In lines 7, 9, and 11 initial conditions of the system are being stored for use on subsequent HOOK calls.

```
C       USER BLOCK 4
C       CHANGE GOV REFERENCE VALUE                                       (1)
        IF((TS.GE.0.0).AND.(TS.LT.0.4) THEN                              (2)
         XLDREF(IREFPT+1) = ORIGVALUE + 0.2                              (3)
        ELSE                                                             (4)
         XLDREF(IREFPT + 1) = ORIGVALUE                                  (5)
        ENDIF                                                            (6)
C       GET MACHINE ANGLE, SPEED, MECHANICAL TORQUE AND GOV OUTPUT       (7)
C       POWER FOR EACH T                                                 (8)
        ANGL = XSTATE(IANG + 2)                                          (9)
        SPNG = XSTATE(IANG + 1)                                          (10)
        MECHPWR = TIN(IMEC + 1)                                          (11)
        GOPWO = XSTATE(IREFPT2 + 2)                                      (12)
C       SUBTRACT INITIAL VALUES OF MECHANICAL TORQUE AND GOV OUTPUT      (13)
C       FOR DOING PRONY ANALYSIS                                         (14)
        DIFMECH = ORIGMECPW - MECHPWR                                    (15)
        DIFGOPW = ORIGGOPW - GOPWO                                       (16)
```

In this block the control actions would be taken. In this case all that is required is to change the governor reference value (XLDREF) by adding 0.2 at TS = 0 and removing 0.2 at TS = 0.4. This is done in lines 1-6. In lines 7-12 some states of both the machine and the governor are being monitored so they can be output to HK.DAT. In lines 15 & 16 the initial value of two of the states are being subtracted so that a signal identification routine (prony) can be run on the data.

```
C       USER BLOCK 5
C       WRITE THE VARIABLES TO DATA FILE HK.DAT                          (1)
        WRITE(21,2000) TS,XLDREF(IREFPT + 1),DIFGOPW,DIFMECH             (2)
2000    FORMAT(F7.4,4X,3(F10.5,4X))                                      (3)
```

The data is written to HK.DAT. This block will only be executed every POUT call to HOOK.

Example 2

In this example HOOK was used to provide a modification of the reference input to a user defined SVC in Figure UDSVC-9 (ETMSP User's MANUAL). As with example 1, the reason that this case was executed was to gather data which was then fed into a numerical algorithm (prony) to identify the transfer function. The base dynamic data file was in this case modified to include a user defined SVC. The added lines are shown below in Table 2. B1 - B7 represent the internal blocks in the user defined SVC. These blocks are lead/lag (LL), washout (WO), gain with limit (GN), and endblock (SD1). In this example case, all the feedback paths were disconnected and only one reference signal was retained. This reference signal was then modified using HOOK.

```
NSVC
PACNW_MN  500  50      0      3
           B1          LL           45.0          1.0          5.0
           B2          LL            1.0          1.0          5.0
           B3          GN            1.0          1.0          0.0
           B4          LL            1.0          0.0          0.02
           B5          SD1         100.0         70.0
           B6          WO            1.0         10.0
           B7          LL            1.0          0.77         0.02

   1    2     2     3     3     4     4     5     6     7     7     1
   1  VREF         -1.0
EDATA
 END
```

Table 2

```
C       USER BLOCK 1
        REAL*4 REFORIG, VVALIN1O, VVALIN5O,                          (1)
     &          VVALIN1, VVALIN5, DIFFVOLT, DIFFBSVC                 (2)
        INTEGER IREFPT, SVCNUM                                       (3)
```

Again, be sure to define all variables that are being used in the user blocks.

```
C       USER BLOCK 2
        POUT = 5                                                     (1)
        NT = 1                                                       (2)
        BSNM(1) = 'PACNW_MN 500'                                     (3)
```

Again, the HOOK will write every 5th call and one bus named PACNW_MN 500 is being monitored.

```
      C       USER BLOCK 3
(1)   C       GIVE THE NUMBER OF SVC
(2)           SVCNUM = 1
(3)   C       GET THE SVC Y-VARIABLE FIELD OFFSET
(4)           IREFPT = JVSVR(SVCNUM)
(5)           GET SVC REFERENCE VALUE AT T=0
(6)           REFORIG = RVREF(IREFPT + 1)
(7)   C       GET INITIAL VALUES OF BUS VOLTAGE AND BSVC
(8)           VVALIN1O = VOLT(PT(1))
(9)           VVALIN5O = VALIN(5)
(10)  C       MAKE ANY INITIAL WRITES TO UNIT 21
(11)          WRITE(21,1000)
(12)  1000 FORMAT('DATA WITH USER DEFINED SVC ON PACNW_MN 500')
```

In lines 2-4 the total number of user defined SVC used in dynamic data file and SVC offset variable are being defined. This offset variable will be used in user block 4 to access the variables.

In lines 6, 8 and 9 the initial conditions of the system are being stored for use on subsequent HOOK calls.

```
      C       USER BLOCK 4
(1)   C       CHANGE SVC REFERENCE VALUE
(2)           IF((TS.GE.0.0).AND.(TS.LT.0.5) THEN
(3)            RVREF(IREFPT+1) = REFORIG + 0.1
(4)           ELSE
(5)            RVREF(IREFPT + 1) = REFORIG
(6)           ENDIF
(7)   C       GET BUS VOLTAGE AND BSVC(SVC SUSCEPTANCE)
(8)           VVALIN1 = VOLT(PT(1))
(9)           VVALIN5 = VALIN(5)
(10)  C       SUBTRACT INITIAL VALUES OF SVC BUS VOLTAGE AND BSVC
(11)  C       FOR DOING PRONY ANALYSIS
(12)          DIFFVOLT = VVOLTIN1 - VVOLTIN1O
(13)          DIFFBSVC = VVOLTIN5 - VVOLTIN5O
```

The user defined SVC reference value (RVREF) is changed by adding 0.1 at TS = 0 and removing 0.1 at TS = 0.5. This is done in lines 1-6. In lines 7-12 SVC bus voltage and BSVC are being monitored so they can be output to HK.DAT. In lines 15 & 16 the initial value of two of the variables are being subtracted so that a signal identification routine (prony) can be run on the data.

```
      C       USER BLOCK 5
(1)   C       WRITE THE VARIABLES TO DATA FILE HK.DAT
(2)           WRITE(21,2000) TS,RVREF(IREFPT + 1),DIFFVOLT,DIFFBSVC
(3)   2000    FORMAT(F7.4,4X,3(F10.5,4X))
```

The data is written to HK.DAT. This block will only be executed every POUT call to HOOK.

When dealing with this user defined SVC it is possible to feedback quite a number of signals through the SVC. These signals are shown in Figure UDSVC-1 (ETMSP User's Manual). This data is available using HOOK and the variable VALIN as mentioned in the list under user defined SVC (TYPE 50). Note, however, that if the dynamic data file consists of two or more user defined SVC's, the sub module DESVC does not store VALIN for each SVC permanently. Therefore, if these data were needed at the end of each iteration, a modification to DESVC would be necessary.

Example 3

In the past two examples HOOK was used to modify references to provide step inputs into the system. It is worth noting that this can also be done using the fault file in ETMSP. In this example, however, HOOK is used to feed a state variable back to a reference input at each iteration of ETMSP. This begins to illustrate the power of HOOK. In this case we close a loop in the system to compare the paths of the closed loop poles with the paths predicted by a root locus plot. The open loop transfer function for the root locus was obtained using the prony transfer function identification algorithm.

A type 1 SVC, which is shown in Figure SVC-1 (ETMSP User's Manual), was added to SOCAL_TX 500 in the base dynamic data file by including the lines shown in Table 3. The difference between phase angles between SOCAL_TX and PACNW_MN was then multiplied by a gain and added to V_{REF}. We were interested in the effect that such a control action would have on the electromechanical modes of the system, although the simulation was done primarily as a check on the prony algorithm.

```
        NSVC
      SOCAL_TX 500    1    0    3
              20.0    1000.0    500.0    1.0    1.0    2.0    2.0    0.2
```

Table 3

The user blocks for example three are shown below.

```
C       USER BLOCK 1                                                              (1)
        REAL*4 REFORIG, VVOLTLO, VVOLT, VVOLTLRO, VVOLTLIO, ANGLELO,              (2)
     &           VVOLTLR, VVOLTLI ,ANGLEL, DIFANGLE, VVOLTI2O,                    (3)
     &           VVOLTR2O,VVOLT2O, ANGLE2O,VVOLTI2, VVOLTR2, ANGLE2,ANGLEO        (4)
        INTEGER IREFPT, SVCNUM, REFL
```

Define all variables.

```
C       USER BLOCK 2
        POUT = 5                                                        (1)
        NT = 1                                                          (2)
        BSNM(1) = 'PACNW_MN 500'                                        (3)
```

HOOK will write every 5th call and one bus named PACNW_MN 500 is being monitored.

```
C       USER BLOCK 3
C       GIVE THE NUMBER OF SVC                                          (1)
        SVCNUM = 1                                                      (2)
C       GET THE SVC Y-VARIABLE FIELD OFFSET                             (3)
        IREFPT = JVSVR(SVCNUM)                                          (4)
C       GET SVC BUS NUMBER                                              (5)
        REFL = NROLD(SVCNUM)                                            (6)
        GET SVC REFERENCE VALUE AT T=0                                  (7)
        REFORIG = RVREF(IREFPT + 1)                                     (8)
C       GET SVC BUS VOLTAGE AT T=0                                      (9)
        VVOLTLO = ABS(VOLT(REFL))                                       (10)
        VVOLTLRO = REAL(VOLT(REFL))                                     (11)
        VVOLTLIO = AIMAG(VOLT(REFL))                                    (12)
C       GET THE PACNW_MN BUS VOLTAGE AT T=0                             (13)
        VVOLT2O = ABS(VOLT(PT(1))                                       (14)
        VVOLTR2O = REAL(VOLT(PT(1))                                     (15)
        VVOLTI2O = AIMAG(VOLT(PT(1))                                    (16)
C       GET THE DIFFERENCE BETWEEN THE PHASE ANGLE OF THE BUSES AT T=0  (17)
        ANGLE2O = (180.0/3.14)*ATAN2(VVOLTI2O,VVOLTR2O)                 (18)
        ANGLELO = (180.0/3.14)*ATAN(VVOLTLIO,VVOLTLRO) - ANGLE2O        (19)
        ANGLEO = ANGLELO                                                (20)
C       MAKE ANY INITIAL WRITES TO UNIT 21                              (21)
        WRITE(21,1000)                                                  (22)
1000    FORMAT('DATA WITH SVC ON SOCAL_TX 500')                         (23)
```

In lines 2-6 the total number of SVC's in dynamic data file and the SVC offset variable are being defined. The offset variable will be used in user block 4 to access the variables.

In lines 8-16 the initial conditions of the system, and in lines 18-20 the initial values of the difference between the phase angles of the buses are being stored for use on subsequent HOOK calls.

```
C       USER BLOCK 4
C       CHANGE SVC REFERENCE VALUE                                      (1)
        IF((TS.GE.0.0).AND.(TS.LT.0.5) THEN                             (2)
         RVREF(IREFPT+1) = REFORIG + 0.1                                (3)
        ELSE                                                            (4)
         RVREF(IREFPT + 1) = REFORIG                                    (5)
        ENDIF
```

```
C      THE DIFFERENCE BETWEEN THE PHASE ANGLES(ANGLEO) WITH GAIN -0.02   (6)
C      IS ADDED AS FEEDBACK TO RVREF                                      (7)
       RVREF(IREFPT + 1) = RVREF(IREFPT +1 ) - .02*ANGLEO                 (8)
       VVOLTL = VVOLTLO - ABS(VOLT(REFL))                                 (9)
       VVOLTLR = REAL(VOLT(REFL))                                         (10)
       VVOLTLI = AIMAG(VOLT(REFL))                                        (11)
       VVOLTR2 = REAL(VOLT(PT(1)))                                        (12)
       VVOLTI2 = AIMAG(VOLT(PT(1))                                        (13)
       ANGLE2 = (180.0/3.14)*ATAN2(VVOLTI2,VVOLTR2)                       (14)
       ANGLEL = (180/3.14)*ATAN2(VVOLTLI,VVOLTLR) - ANGLE2                (15)
C      SUBTRACT INITIAL VALUES OF THE DIFFERENCE BETWEEN PHASE ANGLES     (16)
C      FOR DOING PRONY ANALYSIS                                           (17)
       DIFANGLE = ANGLELO - ANGLEL                                        (18)
       ANGLEO = DIFANGLE                                                  (19)
                                                                          (20)
```

The user defined SVC reference value (RVREF) is changed by adding 0.1 at TS = 0 and removing 0.1 at TS = 0.5 (again, so that prony can get a transfer function). This is done in lines 1-6. In line 8 the difference between the phase angles of the buses is added to the RVREF with gain -0.02. In lines 10-16 the new values of phase angles are calculated and subtracted. In lines 19 and 20 the difference between the phase angles with the initial values are calculated and stored as ANGLEO. ANGLEO is used for feedback in the next iteration.

```
C      USER BLOCK 5
C      WRITE THE VARIABLES TO DATA FILE HK.DAT                            (1)
       WRITE(21,2000) TS,RVREF(IREFPT + 1),VVOLTL,DIFANGLE                (2)
2000   FORMAT(F7.4,4X,3(F10.5,4X))                                        (3)
```

The time, SVC reference, bus voltage at SOCAL_TX 500 and difference in phase angles between PACNW_MN and SOCAL_TX (minus the initial value) are all output. The prony analysis will then be done on DIFANGLE.

Example 4

This example illustrates the use of HOOK to implement a full scale controller on a power system and so illustrates the real power of HOOK. An application of input-state linearization is used to design a control law for the 17 machine power system. The method of input state linearization involves algebraically transforming nonlinear system dynamics into a linear form, so that linear control techniques can be applied. Linearization is achieved by a state transformation and a feedback control law is applied to the machine SOUTH_ G2 which is a steam turbine type generator as shown in Figure 5-67 (ETMSP User's Manual). Below are presented each of the user blocks necessary for adding such a control law.

```
C       USER BLOCK 1
        REAL*4 ASG2O, VLT1, CRNTR, CRNTI, GEPWOLD ,HMM1, DAMP1,TSB1
     &          ASG2, K1, K2, K3, ANG1, ASG1, SPSG2, WD1, GOPWO, GEPWR
     &          DGEP, PM1, UVALUEt
        INTEGER IREFPT1, ISTREF1, ISTREF12, ISTREF13, ISTREF2
     &          IJGREF1, IJGGREF1
```

The variables are defined.

```
C       USER BLOCK 2
        POUT = 5
        NT = 3
        BSNM(1) = 'SOUTH_G220.0'
        BSNM(2) = 'NORTH_G120.0'
        BSNM(3) = 'SOUTH_G120.0'
```

HOOK will write every 5th call and three buses are being monitored.

```
C       USER BLOCK 3
C       GET THE INDICES OF BUSES
        IREFPT1 = MSTATE(5,PT(1))
        ISTREF1 = MSTATE(1,PT(1))
        ISTREF12 = MSTATE(1,PT(1))
        ISTREF13 = MSTAT(1,PT(3))
        ISTREF2 = MYVAR(1,PT(1))
        IJGREF1 = JCONTA(1,PT(1))
        IJGGREF1 = JCONTA(2,PT(1))
C GET THE MACHINE ANGLE, VOLTAGE, CURRENT AND ELECTRICAL POWER AT T=0
        ASG2O = XSTATE(ISTREF1 + 2)
        VLT1 = VOLT(PT(1))
        CRNTR = CDG(1,PT(1))
        CRNTI = CDG(2,PT(1))
        GEPWROLD = PDG(ISTREF2 + 2)
C       GET THE MACHINE INERTIA AND DAMPING FACTOR
        HMM1 = DDAT(IJGREF1 + 1)
        DAMP1 = DDAT(IJGREF1 + 2)
C       GET TURBINE TIME CONSTANT
        TSB1 = DDAT(IJGGREF1 + 9)
C       GIVE GAINS OF LINEAR CONTROLLER
        K1 = -6
        K2 = -11
        K3 = -6
C       MAKE ANY INITIAL WRITES TO UNIT 21
        WRITE(21,1000)
1000    FORMAT('DATA WITH CONTROLLER ON SOUTH_G220.0')
```

In lines 2-8 offset variables are being defined. These index variables can then be used in user block 4 to access the states.

In lines 10-15 initial conditions of the system, and in lines 16-24 machine parameters and controller gains are stored for use on subsequent HOOK calls.

```
C        USER BLOCK 4
C        GET ANGLES AND SPEEDS OF THE MACHINES                                      (1)
         ASG2 = XSTATE(ISTREF1 + 2)                                                 (2)
         ANG1 = XSTATE(ISTREF12 +2)                                                 (3)
         ASG1 = XSTATE(ISTREF13 + 2)                                                (4)
         SPSG2 = XSTATE(ISTREF1 + 1)                                                (5)
C        GET ACCELERATION ,VOLTAGE ,ELECTRICAL POWER AND GOV OUTPUT POWER           (6)
         WD1 = ACCP(ISTREF2 + 6)*HMM1                                               (7)
         VLT1 = VOLT(PT(1))                                                         (8)
         GEPWR = PDG(ISTREF2 + 2)                                                   (9)
         GOPWO = XSTATE(IREFPT1 + 2)                                                (10)
         CRNTR = (CDG(1,PT(1))                                                      (11)
         CRNTI = CDG(2,PT(1))                                                       (12)
C        GET DIFFERENTIATION OF ELECTRICAL POWER                                    (13)
         DGEP = (GEPWR - GEPWROLD)/.01                                              (14)
C        GET GENERATOR MECHANICAL POWER (MECHANICAL TORQUE)                         (15)
         PM1 = TIN(ISTREF2 + 2)                                                     (16)
C .      GET UVALUE (CONTROL LAW) AND ADD IT TO GOV OUTPUT POWER                    (17)
         UVALUE = (1/HMM1)*TSB1*(K1 + DAMP*HMM1)*WD1 + K2*SPSG2                     (18)
       & +K3*(ASG2 - ASG2O) + TSB1*DGEP + (PM1 - 20*GOPWO)                          (19)
         XSTATE(IREFPT1 + 2) = GOPWO + UVALUE/20                                    (20)
         GEPWROLD = GEPWR                                                           (21)
```

In this block a control law is added to the input of the steam turbine of SOUTH_G2 to control the change of machine angle at SOUTH_G2 when a three phase fault is applied at NORTH_G1.

UVALUE, which is calculated in line 18 and 19, is the control law which is added to governor power output(XSTATE(IRFEPT1 + 2).

```
C        USER BLOCK 5
C        WRITE THE VARIABLES TO DATA FILE HK.DAT                                    (1)
         WRITE(21,2000) TS, ASG2, ANG1 , ASG1                                       (2)
2000     FORMAT(F7.4,4X,3(F10.5,4X))                                                (3)
```

CONCLUSION

Our goal in writing HOOK is to create a general interface to ETMSP for implementing any control action on the system that an engineer can dream up. Considering the complexity of ETMSP this is a tall order, and as of now HOOK does not yet fulfill our goal. The bulk of the framework is there, however, which will allow access to all the states and what remains of the task for us to document where all the data is being stored. We have recently received documentation from Ontario Hydro which will be quite helpful in finding all the states and now the primary obstacle in to determine how to generate all the various machine orderings which are used in HOOK.

It should be noted that all of the simulations that we have done using HOOK were conducted using the fourth order Runge-Kutta integrator. If trapazoidal integration is used, the line in SIMULA from which HOOK is called is not executed. This is a solvable problem, but because of our exclusive use of Runge-Kutta, we have not made the fix yet.

APPENDIX B

Sample Codes for the Simulations in Chapter 3

DSYSC.FOR

```
C DSYSC.FOR (VERSION 8/3/92)
C
C A CALLING PROGRAM FOR THE DSYSTEM.FOR SUBROUTINE.
C
C Written By : D. A. Pierre
C Changed By : F. Fatehi
C
C
   IMPLICIT REAL*8 (A-H,O-Z)
   DIMENSION X(16),Y(6),YMO(6),THETA1(200),THETA2(200)
C
C ESTABLISH THE OUTPUT FILES.
C
   OPEN (UNIT=14,NAME='OUTPUT1.DAT',TYPE='NEW')
   OPEN (UNIT=15,NAME='OUTPUT2.DAT',TYPE='NEW')
   OPEN (UNIT=16,NAME='OUTPUT3.DAT',TYPE='NEW')
C
C INITIALIZATION.
C
   N=16
   NY = 6
   DO I=1,N
    X(I) = 0.0
   END DO
c bias is added to as input so u is changed to u2
      u2=u+2
   CALL DSYSTEM(U2,UF,Y,X,WDIS,DEV,N,NY,0,0)
   TIME = 0.0
C
C TIME MARCHES ON.
C
   DO I=1,20001
C
C ASSIGN THE CURRENT CONTROL ACTION.
C
C (IN THIS CASE, COMPARE THE FOLLOWING WITH U = 0.)
C     U = -1
C
C STORE VALUES OF INTEREST.
C
   WRITE(14,102)THETA1(1),THETA1(2),THETA1(3),THETA1(4),THETA1(5)
   WRITE(15,102)THETA1(13),THETA1(14),THETA1(15),THETA1(16)
   WRITE (16,102)TIME,U,Y(3),UNEXT2,Y(5)
 102 FORMAT (5(1x,E12.5))
C
C GET THE NEXT STATE AND OUTPUT VALUES.
C
      DO J=1,6
       YMO(J) = Y(J)
      END DO
```

```
          u2=u+2
          if (i.le.1000)then
        CALL DSYSTEM(U2,UF,Y,X,WDIS,1.,N,NY,1,0)
          elseif (i.le.4000)then
          call dsystem(u2,uf,y,x,wdis,1.,n,ny,2,0)
          elseif (i.le.8000)then
          call dsystem(u2,uf,y,x,wdis,1.,n,ny,1,0)
          elseif (i.le.12000) then
          call dsystem(u2,uf,y,x,wdis,1.,n,ny,2,0)
          elseif (i.le.16000)then
          call dsystem(u2,uf,y,x,wdis,1.,n,ny,1,0)
          else
          call dsystem(u2,uf,y,x,wdis,1.,n,ny,2,0)
          endif
C   CSIG(U(UNEXT)) ARE USED FOR IDENTIFICATION AND OBSEVER
C   CSIG2(U2(UNEXT2)) IS USED AS OLD VALUE FOR CONTROLLER
          CSIG = UNEXT
       CSIG2 = UNEXT
          U = CSIG
          NA=6
          NBB=NA
C CHANGE FOR N1=NA+NBB+NC
          NC=2
          N1=NA+NBB+NC
          XLAMBDA =0.97
          PDIAG = 1.0D6
          ISTART = 1
          IF (IFLAG.EQ.0) THEN
           ISTART = 0
           IFLAG = 1
          ENDIF
C        CALL DSYSTEM(U,UF,YY,XX,WDIS,1.,N,NY,1,0)
          Y1 = Y(1)
          Y2 = Y(2)
C Y3,Y4 ARE YEST TO USE FOR CONTROL ACTION AND OBSERVER
          CALL UUD1(U,Y1,XLAMBDA,N1,NA,NBB,ISTART,QUAL,THETA1,TRACE
     & ,PDIAG,Y3,error1)
          CALL UUD2(U,Y2,XLAMBDA,N1,NA,NBB,ISTART,QUAL,THETA2,TRACE
     & ,PDIAG,Y4,error2)
          IGO=2
          IF (I.LE.201) IGO=1
          IF (I.LE.100) IGO=0
          CALL CONTROL(THETA1,THETA2,Y1,Y2,CSIG,CSIG2,NA,QUAL,IGO,.1,
     & UNEXT,UNEXT2,error1,error2)
c         IF (I.GE.201) UNEXT=0
       TIME = TIME + 0.1
      END DO
      STOP
      END
```

DSYSTEM.FOR

```
C********************************
C
C DSYSTEM.FOR (VERSION 8/3/92)
C
C SUBROUTINE DSYSTEM IS USED TO SIMULATE A DISCRETE-TIME LINEAR SYSTEM
C WITH SCALAR CONTROL U (SUPPLIED BY THE USER) AND WITH DISTURBANCE
C INPUT WDIS (GENERATED WITHIN DSYSTEM). IN ITS PRESENT FORM,
C DSYSTEM.FOR IS GENERAL WITH RESPECT TO THE NUMBER N OF STATE VARIABLES
C AND THE NUMBER NY OF OUTPUT VARIABLES, BUT HAS A SCALAR CONTROL ACTION
C AND A SCALAR DISTURBANCE INPUT. A 4TH-ORDER FILTER FOR THE CONTROL
C ACTION U CREATES UF, A FILTERED CONTROL ACTION FOR USE BY AN IDENTIFIER.
C SIMILARLY FILTERED VERSIONS OF SYSTEM OUTPUTS SHOULD BE INCLUDED IN
C THE SYSTEM MODEL.
C
C FOR INITIALIZATION, PRIOR TO ANY CALLS TO DSYSTEM, AN INITIAL VALUE OF
C THE STATE VECTOR X MUST BE ASSIGNED IN THE CALLING PROGRAM, AS WELL
C AS THE DIMENSION N OF X. AN INITIAL STATE OF 0 IS ASSIGNED INTERNALLY
C FOR THE 4TH-ORDER CONTROL FILTER.
C
C THE FIRST CALL TO DSYSTEM SHOULD BE MADE PRIOR TO ANY ASSIGNMENT
C OF CONTROL ACTIONS, WITH IFLAG1=0 ASSIGNED. THIS WILL RESULT IN
C DSYSTEM READING STATE-SPACE MATRICES FROM FILE 'DSYSTEM.DAT'.
C (THESE MATRICES MUST BE CONSISTENT WITH THE DIMENSION N OF X, AND ARE
C ASSUMED TO BE STORED COLUMN-WISE, WITH AS MANY AS FIVE COLUMNS PER ROW.)
C TWO DIFFERENT SETS OF A.B MATRICES ARE INCLUDED IN ORDER TO PROVIDE FOR
C THE POSSIBILITY OF A STEP CHANGE IN THE SYSTEM DYNAMICS. ONE C MATRIX
C IS ASSUMED TO APPLY FOR BOTH CASES. THE LAST SET OF ENTRIES IN
C DSYSTEM.DAT ARE THE STATE-SPACE MATRICES AB, BB, AND CB THAT
C CHARACTERIZE THE DISCRETE-TIME MODEL OF THE 4TH-ORDER FILTERED
C CONTROL SIGNAL UF.
C
C FOR ALL OTHER CALLS OF DSYSTEM, SET IFLAG1=1 OR IFLAG1=2.
C   WHEN IFLAG1=1, STATE-SPACE MODEL 1 IS USED IN THE COMPUTATIONS.
C   WHEN IFLAG1=2, STATE-SPACE MODEL 2 IS USED IN THE COMPUTATIONS.
C
C IF IFLAG2=0, THE DISTURBANCE 'WDIS' IS NORMAL WHITE NOISE WITH 'DEV'
C FOR ITS STANDARD DEVIATION. IF IFLAG2=1, THE DISTURBANCE IS COLORED
C NOISE, APPROXIMATE '1/f' noise.
C
C Written By : D. A. Pierre
C
C
      SUBROUTINE DSYSTEM(U,UF,Y,X,WDIS,DEV,N,NY,IFLAG1,IFLAG2)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C SET THE PARAMETER NM TO THE MAXIMUM VALUE OF N THAT YOU
C EXPECT TO USE. SET THE PARAMETER NYM TO THE MAXIMUM
C DIMENSION NY OF Y THAT YOU EXPECT TO USE
C
      PARAMETER (NYM=6)
```

```
      PARAMETER (NM=16)
C
      DIMENSION A1(NM,NM),BU1(NM),BD1(NM),C(NYM,NM)
      DIMENSION A2(NM,NM),BU2(NM),BD2(NM)
      DIMENSION Y(1),X(1)
      DIMENSION AB(4,4),BB(4),CB(4),XB(4)
      DIMENSION XTEMP1(NM),XTEMP2(NM)
C
C INITIALIZATION
C
      IF (IFLAG1.EQ.0) THEN
       OPEN (UNIT=8,NAME='DSYSTEM2.DAT',TYPE='OLD')
       IF (N.GT.NM .OR. NY.GT.NYM) THEN
        WRITE (5,100)
        RETURN
       END IF
 100 FORMAT (/,2X,'YOU MUST INCREASE PARAMETERS NM AND/OR',
   &     /,2X,'NYM IN DSYSTEM.FOR.')
       K=1
       L=5
 10    IF (L .GT. N) THEN
        L=N
       END IF
       DO I=1,N
        READ (8,*) (A1(I,J), J=K,L)
       END DO
       IF (L .LT. N) THEN
        K=K+5
        L=L+5
        GO TO 10
       END IF
       READ (8,*) (BU1(I),BD1(I), I=1,N)
       K=1
       L=5
 20    IF (L .GT. N) THEN
        L=N
       END IF
       DO I=1,N
        READ (8,*) (A2(I,J), J=K,L)
       END DO
       IF (L .LT. N) THEN
        K=K+5
        L=L+5
        GO TO 20
       END IF
       READ (8,*) (BU2(I),BD2(I), I=1,N)
       K=1
       L=5
 30    IF (L .GT. N) THEN
        L=N
       END IF
       DO I=1,NY
        READ (8,*) (C(I,J), J=K,L)
```

```
      END DO
      IF (L .LT. N) THEN
       K=K+5
       L=L+5
       GO TO 30
      END IF
      DO I=1,4
       READ (8,*) (AB(I,J), J=1,4)
      END DO
      READ (8,*) (BB(I), I=1,4)
      READ (8,*) (CB(I), I=1,4)
      WDIS = 0.
      DO I=1,4
       XB(I) = 0.0
      END DO
      UF = 0.0
C
C GENERATE INITIAL Y'S
C
      CALL MULTMV1(C,X,Y,NYM,NM,NY,N)
     END IF
C
C A DISTURBANCE INPUT IS GENERATED WITH EVERY CALL TO DSYSTEM.
C
     CALL GNOISE(DIS)
     DIS = DEV*DIS
     IF (IFLAG2 .EQ. 0) THEN
C GAUSSIAN WHITE NOISE CASE; STANDARD DEVIATION = DEV.
      WDIS = DIS
     ELSE
C APPROXIMATE 1/f NOISE (BASED ON T=0.1 SECOND AND ON A
C CORNER FREQUENCY OF 0.1 Hz).
      WDIS = 0.9391*WDIS + 0.0609*DIS
     END IF
C
C
C SYSTEM RESPONSE, AFTER CONTROL IS ASSIGNED IN THE CALLING PROGRAM.
C
     IF (IFLAG1 .NE. 0) THEN
C
C WITH 1ST SYSTEM DYNAMICS ACTIVE:
C
      IF (IFLAG1.EQ.1) THEN
       CALL MULTMV1(A1,X,XTEMP1,NM,NM,N,N)
       CALL MULTVS1(BU1,U,X,N)
       CALL ADDVV1(XTEMP1,X,XTEMP2,N)
       CALL MULTVS1(BD1,WDIS,XTEMP1,N)
       CALL ADDVV1(XTEMP1,XTEMP2,X,N)
       CALL MULTMV1(C,X,Y,NYM,NM,NY,N)
C
C OR WITH 2ND SYSTEM DYNAMICS ACTIVE:
C
      ELSE IF (IFLAG1.EQ.2) THEN
```

```
      CALL MULTMV1(A2,X,XTEMP1,NM,NM,N,N)
      CALL MULTVS1(BU2,U,X,N)
      CALL ADDVV1(XTEMP1,X,XTEMP2,N)
      CALL MULTVS1(BD2,WDIS,XTEMP1,N)
      CALL ADDVV1(XTEMP1,XTEMP2,X,N)
      CALL MULTMV1(C,X,Y,NYM,NM,NY,N)
     END IF
C
C 4TH-ORDER FILTER TO GENERATE UF:
C
     CALL MULTMV1(AB,XB,XTEMP1,4,4,4,4)
     CALL MULTVS1(BB,U,XTEMP2,4)
     CALL ADDVV1(XTEMP1,XTEMP2,XB,4)
     UF = 0.0
     DO I=1,4
      UF = UF + CB(I)*XB(I)
     END DO
    END IF
C
    RETURN
    END
C
C
C*******************************
C VECTOR x SCALAR
C
    SUBROUTINE MULTVS1 (V,S,X,N)
    IMPLICIT REAL*8 (A-H,O-Z)
    DIMENSION V(1),X(1)
    DO I=1,N
     X(I) = S*V(I)
    END DO
    RETURN
    END
C
C
C***************************
C VECTOR + VECTOR
C
    SUBROUTINE ADDVV1 (V1,V2,V3,N)
    IMPLICIT REAL*8 (A-H,O-Z)
    DIMENSION V1(1),V2(1),V3(1)
    DO I=1,N
     V3(I) = V1(I) + V2(I)
    END DO
    RETURN
    END
C
C
C***************************
C MATRIX x VECTOR
C
    SUBROUTINE MULTMV1 (P,X,V,N1,N2,M,N)
```

```
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION P(N1,N2),X(N2),V(N1)
      DO I=1,M
        V(I) = 0.0
      END DO
      DO I=1,M
        DO J=1,N
          V(I) = V(I) + P(I,J)*X(J)
        END DO
      END DO
      RETURN
      END
C
C
C*****************************
C
C
C       GNOISE procuces quasi-Gaussian noise samples from
C       a population with zero mean and unit variance.
C       It sums twelve results from the pseudorandom number
C       generator RAN(I1,I2) (which produces results uniformly
C       distributed on the unit interval, with mean 0.5 and
C       variance of 1/12) to give a near-Gaussian distribution
C       with mean of 6.0 and unit variance. Then 6.0 is
C       subtracted to give the result. The seeds of RAN are
C       two-byte, large, odd integers.
C
C
        SUBROUTINE GNOISE(X)
      IMPLICIT REAL*8 (A-H,O-Z)
        INTEGER*2 I1,I2
C
        DATA I1,I2 /9587,7941/
C
        X = 0.0
        DO I=1,12
          X = X + RAN(I1,I2)
        END DO
C
C Reset mean of distribution
        X = X - 6.0
C
        RETURN
        END
C
C
C*****************************
```

DSYSTEM.DAT

8.7839330e-001	9.5415438e-002	4.7988568e-002	1.6213645e-003	0.0000000e+000
-2.3691723e+000	8.6885175e-001	9.2172708e-001	4.7826431e-002	0.0000000e+000
4.7988568e-002	1.6213645e-003	9.0238758e-001	9.6226120e-002	0.0000000e+000
9.2172708e-001	4.7826431e-002	-1.9083088e+000	8.9276497e-001	0.0000000e+000
-1.6587904e+000	8.6508049e-001	6.5137497e-001	2.6764444e-002	4.3282798e-002
-3.0147245e-002	2.3551370e-002	1.1896188e-002	4.0279879e-004	0.0000000e+000
-1.0957756e+001	8.0820224e+000	4.3311026e+000	1.3902494e-001	7.5901555e-001
-3.5078653e-001	4.3961024e-001	1.3934994e-001	3.2499925e-003	2.7353721e-001
-1.2449707e+000	1.8664636e+000	4.9536239e-001	9.9993822e-003	1.4809092e+000
-2.5130634e-002	4.9864473e-002	1.0015794e-002	1.6411671e-004	6.0801749e-002
6.5137497e-001	2.6764444e-002	-1.3331029e+000	8.7846271e-001	0.0000000e+000
1.1896188e-002	4.0279879e-004	-2.4199151e-002	2.3752769e-002	0.0000000e+000
4.3311026e+000	1.3902494e-001	-8.7922049e+000	8.1515349e+000	0.0000000e+000
1.3934994e-001	3.2499925e-003	-2.8111155e-001	4.4123524e-001	0.0000000e+000
4.9536239e-001	9.9993822e-003	-9.9728946e-001	1.8714633e+000	0.0000000e+000
1.0015794e-002	1.6411671e-004	-2.0122737e-002	4.9946531e-002	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
-9.3950727e-001	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
9.7530991e-001	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
-8.4767700e+000	-1.7346463e-001	-9.3480838e+000	0.0000000e+000	0.0000000e+000
-4.4919312e-001	3.6515952e-002	2.7370973e-001	0.0000000e+000	0.0000000e+000
-1.8965807e+000	2.2335635e-001	2.0899143e+000	-1.2981058e-001	-5.4799985e+000
-5.0377048e-002	1.3242516e-002	3.8552421e-001	2.1406244e-002	5.0304362e-001
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
4.3282798e-002	-9.3950727e-001	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	9.7530991e-001	0.0000000e+000	0.0000000e+000	0.0000000e+000
7.5901555e-001	-8.4767700e+000	-1.7346463e-001	-9.3480838e+000	0.0000000e+000
2.7353721e-001	-4.4919312e-001	3.6515952e-002	2.7370973e-001	0.0000000e+000
1.4809092e+000	-1.8965807e+000	2.2335635e-001	2.0899143e+000	-1.2981058e-001
6.0801749e-002	-5.0377048e-002	1.3242516e-002	3.8552421e-001	2.1406244e-002
0.0000000e+000				
0.0000000e+000				

0.0000000e+000
0.0000000e+000
0.0000000e+000
0.0000000e+000
0.0000000e+000
0.0000000e+000
0.0000000e+000
0.0000000e+000
0.0000000e+000
0.0000000e+000
0.0000000e+000
0.0000000e+000
-5.4799985e+000
5.0304362e-001
4.8806210e-003 4.0882085e-005
9.5415438e-002 1.6213645e-003
4.0882085e-005 4.9010620e-003
1.6213645e-003 9.6226120e-002
6.6655146e-002 7.5882435e-004
1.2099576e-003 1.0169378e-005
4.3961024e-001 3.2499925e-003
1.4055578e-002 6.0291729e-005
4.9864473e-002 1.6411671e-004
1.0061368e-003 2.2787266e-006
7.5882435e-004 6.7034558e-002
1.0169378e-005 1.2150422e-003
3.2499925e-003 4.4123524e-001
6.0291729e-005 1.4085724e-002
1.6411671e-004 4.9946531e-002
2.2787266e-006 1.0072762e-003
9.5098938e-001 9.7859789e-002 4.8599749e-002 1.6336403e-003 0.0000000e+000
-9.6226149e-001 9.4120340e-001 9.4592509e-001 4.8436385e-002 0.0000000e+000
4.8599749e-002 1.6336403e-003 9.0238963e-001 9.6226149e-002 0.0000000e+000
9.4592509e-001 4.8436385e-002 -1.9081866e+000 8.9276701e-001 0.0000000e+000
-6.7034567e-001 9.0547439e-001 6.6271370e-001 2.7011020e-002 4.3282798e-002
-1.2150423e-002 2.4158632e-002 1.2048218e-002 4.0585493e-004 0.0000000e+000
-4.4123526e+000 8.2914628e+000 4.3797133e+000 1.3992608e-001 7.5901555e-001
-1.4085724e-001 4.4449919e-001 1.4025247e-001 3.2639285e-003 2.7353721e-001
-4.9946532e-001 1.8814968e+000 4.9782003e-001 1.0033478e-002 1.4809092e+000
-1.0072762e-002 5.0111060e-002 1.0049931e-002 1.6452834e-004 6.0801749e-002
6.6271370e-001 2.7011020e-002 -1.3330594e+000 8.7846337e-001 0.0000000e+000
1.2048218e-002 4.0585493e-004 -2.4198641e-002 2.3752777e-002 0.0000000e+000
4.3797133e+000 1.3992608e-001 -8.7920659e+000 8.1515367e+000 0.0000000e+000
1.4025247e-001 3.2639285e-003 -2.8110971e-001 4.4123526e-001 0.0000000e+000
4.9782003e-001 1.0033478e-002 -9.9728535e-001 1.8714633e+000 0.0000000e+000
1.0049931e-002 1.6452834e-004 -2.0122693e-002 4.9946532e-002 0.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
-9.3950727e-001 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
9.7530991e-001 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
-8.4767700e+000 -1.7346463e-001 -9.3480838e+000 0.0000000e+000 0.0000000e+000

-4.4919312e-001	3.6515952e-002	2.7370973e-001	0.0000000e+000	0.0000000e+000
-1.8965807e+000	2.2335635e-001	2.0899143e+000	-1.2981058e-001	-5.4799985e+000
-5.0377048e-002	1.3242516e-002	3.8552421e-001	2.1406244e-002	5.0304362e-001
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000	0.0000000e+000
4.3282798e-002	-9.3950727e-001	0.0000000e+000	0.0000000e+000	0.0000000e+000
0.0000000e+000	9.7530991e-001	0.0000000e+000	0.0000000e+000	0.0000000e+000
7.5901555e-001	-8.4767700e+000	-1.7346463e-001	-9.3480838e+000	0.0000000e+000
2.7353721e-001	-4.4919312e-001	3.6515952e-002	2.7370973e-001	0.0000000e+000
1.4809092e+000	-1.8965807e+000	2.2335635e-001	2.0899143e+000	-1.2981058e-001
6.0801749e-002	-5.0377048e-002	1.3242516e-002	3.8552421e-001	2.1406244e-002
0.0000000e+000				
0.0000000e+000				
0.0000000e+000				
0.0000000e+000				
0.0000000e+000				
0.0000000e+000				
0.0000000e+000				
0.0000000e+000				
0.0000000e+000				
0.0000000e+000				
0.0000000e+000				
0.0000000e+000				
0.0000000e+000				
0.0000000e+000				
-5.4799985e+000				
5.0304362e-001				
4.9421499e-003	4.1087500e-005			
9.7859789e-002	1.6336403e-003			
4.1087500e-005	4.9010624e-003			
1.6336403e-003	9.6226149e-002			
6.7797763e-002	7.6319617e-004			
1.2252629e-003	1.0220549e-005			
4.4449919e-001	3.2639285e-003			
1.4146201e-002	6.0476855e-005			
5.0111060e-002	1.6452834e-004			
1.0095593e-003	2.2831287e-006			
7.6319617e-004	6.7034567e-002			
1.0220549e-005	1.2150423e-003			

```
3.2639285e-003 4.4123526e-001
6.0476855e-005 1.4085724e-002
1.6452834e-004 4.9946532e-002
2.2831287e-006 1.0072762e-003
0.0000000e+000 1.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 1.0000000e+000 0.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 1.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 1.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
1.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
0.0000000e+000
0.0000000e+000
0.0000000e+000
0.0000000e+000
0.0000000e+000
1.0000000e+000
-1.7346463e-001 -9.3480838e+000 0.0000000e+000 0.0000000e+000
3.6515952e-002 2.7370973e-001 0.0000000e+000 0.0000000e+000
2.2335635e-001 2.0899143e+000 -1.2981058e-001 -5.4799985e+000
1.3242516e-002 3.8552421e-001 2.1406244e-002 5.0304362e-001
9.3480838e+000
7.2629027e-001
3.3900842e+000
1.1143217e-001
0.0000000e+000 0.0000000e+000 0.0000000e+000 1.0000000e+000
```

UUD.FOR

```
c_____________________________________________________________
c
c      uud
c
c      Least-Squares estimation of the parameters, using UD
c      factorization. The plant can be represented by the following
c      equation :
c
c      y(k)=-a(1)y(k-1)-...-a(na)y(k-na)+
c           b(1)u(k-1)+...+b(nb)u(k-nb)+
c           c(1)e(k-1)+...+c(nc)e(k-nc)
c
c      Where :
c      y: Is the output of the system.
c      u: Is the input to the system.
c      e: Is the error in measurements.
c
c
c      Input :
c      u: New input measurement.
c      y: New output measurement.
c      xlambda: Forgetting factor.
c      n: i.e. n = na + nb + nc
c      na: Number of a coeff.
c      nb: Number of b coeff.
c      theta: A prior estimate of the parameters.
c      istart: 0 => Initialize all the variables. This is
c           the first pass through this subroutine.
c        1 => Continue with estimation.
c        2 => Stope estimation.
c      pdiag: Initial value for diagonal terms of covariance
c        matrix.
c
c
c      Output :
c      theta: Updated version of estimate vector.
c      quality: Quality of the estimation.
c        1 => perfect estimation.
c        0 => very poor estimation.
c      trace: Trace of the covariance matrix.
c
c      Written By : Iraj Sadighi
c      7-20-1987
c  Changed By : F. Fatehi
c  10-20-1992
c_____________________________________________________________
```

```
      subroutine uud1(u,y,xlambda,n,na,nb,istart,quality,THETA,
     $     TRACE,pdiag,Y3,error1)
      implicit real*8 (a-h , o-z)
      implicit integer (i-n)

      dimension theta(1),fi(200),diag(200),offdiag(200)
      dimension xk(200),ud(200),utfi(200),fii(200)
      if ( istart .eq. 2 ) goto 1010
      nc = n-na-nb
C     n = na+nb
      if ( istart .eq. 0 ) then
c
c     Initialization.
c
      nup = n*(n-1)/2
      do 100 i =1,nup
      theta(i)  = 0.0
      fi(i)     = 0.0
      fii(i)  = 0.0
      offdiag(i) = 0.0
      ud(i)     = 0.0
      utfi(i)   = 0.0
c
c     let diag vector be a big number which means the initial
c     values are not accurate.
c
      diag(i)   = pdiag
 100  continue
      ydc=0.0
      udc=0.0
      alphaj = 0.01
c     istart = 1
      end if
c
c     estimate the D.C. bias from the measurements.
c
C     ydc = xlambda*ydc + (1.0 - xlambda)*y
C     udc = xlambda*udc + (1.0 - xlambda)*u
C     y = y - ydc
C     u = u - udc
C
C     normalization.
c
c     finorm = 1.0
c     do 4444 i =1,n
c     fitemp = abs(fi(i))
c     if(fitemp .gt. finorm) finorm=fitemp
c 4444continue
c
c     if(finorm .gt. 1.0) then
c     do 4445 i=1,n
c     fi(i) = fi(i)/finorm
```

```
c 4445continue
c       y = y/finorm
c       end if
c
c
c       Estimate the error. i.e. error = act.y - est.y
c
        error = y
        do 11 i =1,n
        error = error - fi(i)*theta(i)
 11     continue
c
c       find an estimate to y (esty) when no noise is presented.
c
        esty=0.0
        do 864 i=1 ,na+nb
        esty = esty + fi(i)*theta(i)
 864    continue
C ADDED BY FF
        Y3=ESTY
        error1=error
c
c       find an estimate to the disturbance term.
c
c       estd = y-esty
c
        estd = error
c
c       estd = 0.0
c       do 5874 i=na+nb+1,n
c       estd = estd + fi(i)*theta(i)
c 5874continue
c


c
c       find (u**T)*fi
c
        do 2121 i=1,n
        utfi(i) = fi(i)
        do 2122 j=1,i-1
        k = i*(i-3)/2+1+j
        utfi(i) = utfi(i) + fi(j)*offdiag(k)
 2122   continue
 2121   continue
c
c       find u*d
c
        do 2211 i=1,n
        k = i*(i+1)/2
        ud(k) = diag(i)
        do 2212 j=i+1,n
        k=j*(j-3)/2+1+i
```

```
      m=j*(j-1)/2+i
      ud(m)=diag(j)*offdiag(k)
2212  continue
2211  continue

c
c     find the trace of covariance matrix.
c
      trace = 0.0
      do 12 i = 1,n
      trace = trace + diag(i)
      do 1212 j = i+1,n
      k = j*(j-3)/2+1+i
      trace = trace + diag(j)*offdiag(k)*offdiag(k)
1212  continue
12    continue
c
c     if trace of p < 1.0 and trace of p >initial value then
c     reset the p matrix.
c
      if((trace .gt. 1000000.0) .or. (trace .lt. 1.0)) then
      do 1457 i =1,n
      diag(i) = 10000.0
1457  continue
      end if

c
c     find a new forgetting factor.
c
      r = 1.0 + alphaj - xlambda
c     r=0.0
c     do 5974 i=1,n
c     r = r + utfi(i)*utfi(i)*diag(i)
c 5974continue
c
      xnorm = 0.0
      do 10 i = 1,n
      xxnorm = 0.0
      do 111 j = i,n
      k=j*(j-1)/2+i
      xxnorm = xxnorm + ud(k)*utfi(j)
111   continue
      xnorm = xnorm + xxnorm*xxnorm
10    continue
      rp = r*r - 4.0*(xnorm)/trace
      if (rp .ge. 0.0) then
      xlambda = 1.0 - (r - sqrt(rp))/2.0
      if (xlambda .lt. 0.0) xlambda=0.5
      if (xlambda .ge. 1.0) xlambda=0.97
      end if
```

```
c
c       write the results to the data file.
c
c
c       find the quality of the estimates.
c
        if((abs(y) .gt. 1.0e-10) .and. (abs(y) .gt. abs(error))) then
        quality = 1.0 - abs(error/(y-ydc))
        if(quality .gt. 1.0) quality=1.0
        if(quality .lt. 0.0) quality=0.0
        end if

c
c       Start the procedure.
c
        xperr = error

c
c       Calculate gain and covariance using U-D method.
c
        fj = fi(1)
        vj = diag(1)*fj
        xk(1) = vj
        alphaj = 1.0 + vj*fj
        diag(1) = diag(1)/alphaj/xlambda
        if ( n .gt. 1 ) then
        kf = 0
        ku = 0
        do 20 j =2,n
        fj = fi(j)
        do 30 i =1,j-1
        kf = kf+1
         fj = fj + fi(i)*offdiag(kf)
30      continue
        vj = fj*diag(j)
        xk(j) = vj
        ajlast = alphaj
        alphaj = ajlast + vj*fj

        airaj = ajlast/(alphaj*xlambda)
        if(airaj .gt. 1.0) airaj=1.0
        diag(j) = diag(j) * airaj

c       diag(j) = diag(j) * ajlast/(alphaj*xlambda)
        pj = -fj/ajlast
        do 40 i =1,j-1
        ku = ku + 1
        w = offdiag(ku) + xk(i)*pj
        xk(i) = xk(i) + offdiag(ku)*vj
        offdiag(ku) = w
```

```
40    continue
20    continue
      endif

c
c     Update parameter estimates.
c
      do 50 i =1,n
      theta(i) = theta(i) + xperr*xk(i)/alphaj
50    continue

c
c     write the estimate of the coeff. out.
c
c     if(finorm .gt. 1.0) then
c     y=y*finorm
c     do 53 i=1,n
c     fi(i) = fi(i)*finorm
c 53  continue
c     end if
c
c
c     Udate the fi or data vector.
c
      do 60 i =1,n-1
      fi(n+1-i) = fi(n-i)
60    continue
      fi(1) = -y
      fi(na+1) = u
      fi(na+nb+1) = estd
c     n=na+nb+nc
      do 61 i =1,n-1
      fii(n+1-i) = fii(n-i)
61    continue
      fii(1) = -acty
      fii(na+1) = u
      fii(na+nb+1)=eee

c
c     return to calling program.
c
1010  return
      end
      subroutine uud2(u,y,xlambda,n,na,nb,istart,quality,THETA,
   $      TRACE,pdiag,Y4,error2)
      implicit real*8 (a-h , o-z)
      implicit integer (i-n)

      dimension theta(1),fi(200),diag(200),offdiag(200)
      dimension xk(200),ud(200),utfi(200),fii(200)
      if ( istart .eq. 2 ) goto 1010
```

```
      nc = n-na-nb
C     n = na+nb
      if ( istart .eq. 0 ) then
c
c     Initialization.
c
      nup = n*(n-1)/2
      do 100 i =1,nup
      theta(i)  = 0.0
      fi(i)     = 0.0
      fii(i)  = 0.0
      offdiag(i) = 0.0
      ud(i)     = 0.0
      utfi(i)   = 0.0
c
c     let diag vector be a big number which means the initial
c     values are not accurate.
c
      diag(i)   = pdiag
 100  continue
      ydc=0.0
      udc=0.0
      alphaj = 0.01
c     istart = 1
      end if
c
c     estimate the D.C. bias from the measurements.
c
C     ydc = xlambda*ydc + (1.0 - xlambda)*y
C     udc = xlambda*udc + (1.0 - xlambda)*u
C     y = y - ydc
C     u = u - udc
C
C     normalization.
c
c     finorm = 1.0
c     do 4444 i =1,n
c     fitemp = abs(fi(i))
c     if(fitemp .gt. finorm) finorm=fitemp
c 4444continue
c
c     if(finorm .gt. 1.0) then
c     do 4445 i=1,n
c     fi(i) = fi(i)/finorm
c 4445continue
c     y = y/finorm
c     end if
c
c
c     Estimate the error. i.e. error = act.y - est.y
c
      error = y
```

```
      do 11 i =1,n
      error = error - fi(i)*theta(i)
 11   continue
c
c     find an estimate to y (esty) when no noise is presented.
c
      esty=0.0
      do 864 i=1 ,na+nb
      esty = esty + fi(i)*theta(i)
 864  continue
C  ADDED BY FF
      Y4=ESTY
      error2=error
c
c     find an estimate to the disturbance term.
c
c     estd = y-esty
c
      estd = error
c
c     estd = 0.0
c     do 5874 i=na+nb+1,n
c     estd = estd + fi(i)*theta(i)
c 5874continue
c


c
c     find (u**T)*fi
c
      do 2121 i=1,n
      utfi(i) = fi(i)
      do 2122 j=1,i-1
      k = i*(i-3)/2+1+j
      utfi(i) = utfi(i) + fi(j)*offdiag(k)
 2122 continue
 2121 continue
c
c     find u*d
c
      do 2211 i=1,n
      k = i*(i+1)/2
      ud(k) = diag(i)
      do 2212 j=i+1,n
      k=j*(j-3)/2+1+i
      m=j*(j-1)/2+i
      ud(m)=diag(j)*offdiag(k)
 2212 continue
 2211 continue
```

```
c
c       find the trace of covariance matrix.
c
        trace = 0.0
        do 12 i = 1,n
        trace = trace + diag(i)
        do 1212 j = i+1,n
        k = j*(j-3)/2+1+i
        trace = trace + diag(j)*offdiag(k)*offdiag(k)
 1212   continue
 12     continue
c
c       if trace of p < 1.0 and trace of p >initial value then
c       reset the p matrix.
c
        if((trace .gt. 1000000.0) .or. (trace .lt. 1.0)) then
        do 1457 i =1,n
        diag(i) = 10000.0
 1457   continue
        end if


c
c       find a new forgetting factor.
c
        r = 1.0 + alphaj - xlambda
c       r=0.0
c       do 5974 i=1,n
c       r = r + utfi(i)*utfi(i)*diag(i)
c 5974continue
c
        xnorm = 0.0
        do 10 i = 1,n
        xxnorm = 0.0
        do 111 j = i,n
        k=j*(j-1)/2+i
        xxnorm = xxnorm + ud(k)*utfi(j)
 111    continue
        xnorm = xnorm + xxnorm*xxnorm
 10     continue
        rp = r*r - 4.0*(xnorm)/trace
        if (rp .ge. 0.0) then
        xlambda = 1.0 - (r - sqrt(rp))/2.0
        if (xlambda .lt. 0.0) xlambda=0.5
        if (xlambda .ge. 1.0) xlambda=0.97
        end if
c
c       write the results to the data file.
c
```

```
c
c     find the quality of the estimates.
c
      if((abs(y) .gt. 1.0e-10) .and. (abs(y) .gt. abs(error))) then
      quality = 1.0 - abs(error/(y-ydc))
      if(quality .gt. 1.0) quality=1.0
      if(quality .lt. 0.0) quality=0.0
      end if

c
c     Start the procedure.
c
      xperr = error

c
c     Calculate gain and covariance using U-D method.
c
      fj = fi(1)
      vj = diag(1)*fj
      xk(1) = vj
      alphaj = 1.0 + vj*fj
      diag(1) = diag(1)/alphaj/xlambda
      if ( n .gt. 1 ) then
      kf = 0
      ku = 0
      do 20 j =2,n
      fj = fi(j)
      do 30 i =1,j-1
      kf = kf+1
       fj = fj + fi(i)*offdiag(kf)
30    continue
      vj = fj*diag(j)
      xk(j) = vj
      ajlast = alphaj
      alphaj = ajlast + vj*fj

      airaj = ajlast/(alphaj*xlambda)
      if(airaj .gt. 1.0) airaj=1.0
      diag(j) = diag(j) * airaj

c     diag(j) = diag(j) * ajlast/(alphaj*xlambda)
      pj = -fj/ajlast
      do 40 i =1,j-1
      ku = ku + 1
      w = offdiag(ku) + xk(i)*pj
      xk(i) = xk(i) + offdiag(ku)*vj
      offdiag(ku) = w
40    continue
20    continue
      endif
```

```
c
c     Update parameter estimates.
c
      do 50 i =1,n
      theta(i) = theta(i) + xperr*xk(i)/alphaj
 50   continue

c
c     write the estimate of the coeff. out.
c
c     if(finorm .gt. 1.0) then
c     y=y*finorm
c     do 53 i=1,n
c     fi(i) = fi(i)*finorm
c 53  continue
c     end if
c
c
c     Udate the fi or data vector.
c
      do 60 i =1,n-1
      fi(n+1-i) = fi(n-i)
 60   continue
      fi(1) = -y
      fi(na+1) = u
      fi(na+nb+1) = estd
c     n=na+nb+nc
      do 61 i =1,n-1
      fii(n+1-i) = fii(n-i)
 61   continue
      fii(1) = -acty
      fii(na+1) = u
      fii(na+nb+1)=eee

c
c     return to calling program.
c
 1010 return
      end
```

CONTROL.FOR

```
C*************************************
C  THIS SUBROUTINE MAKES A DECISION ON THE TYPE OF CONTROL
C  ACTION AND THEN CREATES THE CONTROL ACTION.
C
C  INPUTS:
C     THETA1CHARACTERISTIC POLYNOMIAL OF THE PLANT, FOR OUTPUT NUMBER 1.
C     THETA2    //      //   ////  //  //  //  // 2.
C     B1CONTROL INPUT VECTOR OF THE PLANT, FOR OUTPUT NUMBER 1.
C     B2 //  //  // // // // // //  // 2.
C     Y1PLANT OUTPUT NUMBER 1.
C     Y2PLANT OUTPUT NUMBER 2.
C     UPLANT INPUT.
C     NSYSTEM ORDER.
C     QUALQUALITY OF A AND B COEFFICIENTS.(0-1)
C     IGOFLAG FOR ADAPTIVE CONTROL.
C     TTIME.
C
C  OUTPUT:
C     UNEXTCONTROL ACTION.
C
C  WRITTEN BY: D. TRUDNOWSKi
C  CHANGED BY: F. FATEHI
C*************************************

      SUBROUTINE CONTROL(THETA1,THETA2,Y1,Y2,U,U2,N,QUAL,IGO,T,
     & UNEXT,UNEXT2,error1,error2)
      IMPLICIT NONE
      REAL*8 THETA1(1),THETA2(1)
      REAL*8 A1(10),A2(10),B1(10),B2(10),C1(10),C2(10),L1(10),L2(10)
      REAL*8  R11(10,10), R12(10,10), R21(10,10), R22(10,10)
      REAL*8 TR11(10,10),TR12(10,10),TR21(10,10),TR22(10,10)
      REAL*8 XHAT1(10),XHAT2(10),H1(10),H2(10),GZERO1(10),GZERO2(10)
      REAL*8 ALPHA,U,Y1,Y2,QUAL,T,UNEXT,UPROB,FIRST,UNEXT2,U2
      REAL*8 W1,W2,W12,G1(10),G2(10),gammaY1,gammaY2,PAR1(10),PAR2(10)
      real*8 error1,error2
      INTEGER PFLG,I,J,ICOUNT,IGO,N,N2,NU,NA,NB,NM

C*******************************
C  FIRST TIME THROUGH INITIALIZE MATRICES
C*******************************
        IF (ICOUNT.GE.80) ICOUNT=0
      IF (FIRST.EQ.0.0) THEN
       PFLG = 0
       DO I=1,N
        DO J=1,N
         R11(I,J) = 0.0
           R12(I,J) = 0.0
           R21(I,J) = 0.0
           R22(I,J) = 0.0
```

```
      TR11(I,J) = 0.0
        TR12(I,J) = 0.0
        TR21(I,J) = 0.0
        TR22(I,J) = 0.0
     END DO
       A1(I)   = 0.0
       A2(I)   = 0.0
       B1(I)   = 0.0
       B2(I)   = 0.0
       C1(I)   = 0.0
       C2(I)   = 0.0
     H1(I)   = 0.0
       H2(I)   = 0.0
     G1(I)   = 0.0
       G2(I)   = 0.0
       XHAT1(I) = 0.0
       XHAT2(I) = 0.0
       GZERO1(I) = 0.0
       GZERO2(I) = 0.0
     END DO
C ADDED INITIALIZATION
      W1 = .05
      W2 = .1
      W12 = W1 + W2
      N2 = -1
      NU = 5
      gammaY1 = 1
      gammaY2 = 1

C     PAR1 = THETA1
C       PAR2 = THETA2
      NA = N
      NB = N
      R11(1,1) = 1.0
     R22(1,1) = 1.0
      ALPHA  = 0.0
     FIRST  = 1.0
     ICOUNT = 0
    END IF
C********************************
C
C     UPDATE A1, A2, B1 AND B2 VECTORS.
C
C********************************
   DO I=1,N
    A1(I) = -THETA1(I)
    B1(I) =  THETA1(N+I)
      C1(I) =  THETA1(2*N+I)
    A2(I) = -THETA2(I)
    B2(I) =  THETA2(N+I)
      C2(I) =  THETA2(2*N+I)
   ENDDO
```

```
C*******************************
C   IF THE FLAG IGO IS 0 THEN THE CONTROL ACTION IS NOT
C   DETERMINED BY THE ADAPTIVE CONTROL
C********************************
      UPROB =0.4
      IF (IGO.EQ.0) THEN

       UNEXT =  UPROB
       IF (ICOUNT .LE. 73) UNEXT = -UPROB
       IF (ICOUNT .LE. 65) UNEXT =  UPROB
       IF (ICOUNT .LE. 61) UNEXT = -UPROB
       IF (ICOUNT .LE. 54) UNEXT =  UPROB
       IF (ICOUNT .LE. 46) UNEXT = -UPROB
       IF (ICOUNT .LE. 42) UNEXT =  UPROB
       IF (ICOUNT .LE. 34) UNEXT = -UPROB
       IF (ICOUNT .LE. 27) UNEXT =  UPROB
       IF (ICOUNT .LE. 23) UNEXT = -UPROB
       IF (ICOUNT .LE. 16) UNEXT =  UPROB
       IF (ICOUNT .LE. 13) UNEXT = -UPROB
       IF (ICOUNT .LE. 5)  UNEXT =  UPROB
       ICOUNT = ICOUNT +1
       RETURN
      ENDIF
      IF (IGO .GE. 1) THEN
C********************************
C   START THE ADAPTIVE CONTROL
C********************************
      CALL RICTWO(THETA1,THETA2,W1,W2,W12,G1,G2,ALPHA,
     &      R11,R12,R22,NA,NB,N2,NU,
     &     gammaY1,gammaY2)
      CALL OBSERV(B1,B2,A1,A2,C1,C2,error1,error2,H1,H2,
     &      XHAT1,XHAT2,U,Y1,Y2,N,T)
       IF (IGO .EQ. 1) THEN

       UNEXT =  UPROB
       IF (ICOUNT .LE. 73) UNEXT = -UPROB
       IF (ICOUNT .LE. 65) UNEXT =  UPROB
       IF (ICOUNT .LE. 61) UNEXT = -UPROB
       IF (ICOUNT .LE. 54) UNEXT =  UPROB
       IF (ICOUNT .LE. 46) UNEXT = -UPROB
       IF (ICOUNT .LE. 42) UNEXT =  UPROB
       IF (ICOUNT .LE. 34) UNEXT = -UPROB
       IF (ICOUNT .LE. 27) UNEXT =  UPROB
       IF (ICOUNT .LE. 23) UNEXT = -UPROB
       IF (ICOUNT .LE. 16) UNEXT =  UPROB
       IF (ICOUNT .LE. 13) UNEXT = -UPROB
       IF (ICOUNT .LE. 5)  UNEXT =  UPROB
       ICOUNT = ICOUNT + 1
       IF(ICOUNT .GT. 80) THEN
        ICOUNT = 0
        UPROB = ABS(UPROB) + 0.08
       ENDIF
```

```
c        WRITE(16,601) T,(G1(J),J=1,N),ALPHA
C         WRITE(*,601) T,(G2(J),J=1,N),ALPHA
 601     format(8(1X,1Pe10.3))
       RETURN
       ENDIF
      ENDIF
C*********************************
C   CALCULATE THE CONTROL SIGNAL
C*********************************
C  U2 (UNEXT2(UNPROBED)) IS USED FOR MAKING UNEXT2 THEN PROB IS ADDED
C  (UNEXT) WHICH IS USED FOR IDENTIFIER AND OBSERVER . UNEXT2 IS USED
C   FOR NEXT CALCULATION OF CONTROL SIGNAL AS OLD VALUE
      IF (IGO .EQ. 2) THEN
       UNEXT = ALPHA * U2
       DO J=1,N
         UNEXT = UNEXT - G1(J)*XHAT1(J) - G2(J)*XHAT2(J)
       END DO
      ENDIF
c     WRITE(16,602) T,(G1(J),J=1,N),ALPHA
C      WRITE(*,602) T,(G2(J),J=1,N),ALPHA
 602  format(8(1X,1Pe10.3))

       IF (UNEXT.GT.8) THEN
         UNEXT = 8
       ELSE IF (UNEXT.LT.-8) THEN
         UNEXT = -8
       END IF
          UNEXT2 = UNEXT
        IF (ICOUNT .LE. 73) UNEXT =UNEXT -UPROB
        IF (ICOUNT .LE. 65) UNEXT = UNEXT+ UPROB
        IF (ICOUNT .LE. 61) UNEXT =UNEXT -UPROB
        IF (ICOUNT .LE. 54) UNEXT = UNEXT+ UPROB
        IF (ICOUNT .LE. 46) UNEXT =UNEXT -UPROB
        IF (ICOUNT .LE. 42) UNEXT =UNEXT+  UPROB
        IF (ICOUNT .LE. 34) UNEXT =UNEXT -UPROB
        IF (ICOUNT .LE. 27) UNEXT = UNEXT+ UPROB
        IF (ICOUNT .LE. 23) UNEXT =UNEXT -UPROB
        IF (ICOUNT .LE. 16) UNEXT =UNEXT+  UPROB
        IF (ICOUNT .LE. 13) UNEXT = UNEXT-UPROB
        IF (ICOUNT .LE. 5)  UNEXT = UNEXT+ UPROB
        ICOUNT = ICOUNT + 1
        IF(ICOUNT .GT. 80) THEN
          ICOUNT = 0
          ENDIF
      RETURN
      END
C
C
     SUBROUTINE RICTWO(PAR1,PAR2,W1,W2,W12,G1,G2,
    &             ALPHA,
    &             R11,R12,R22,NA,NB,
```

```
     &          N2,NU,gammaY1,gammaY2)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION B1(10),B2(10),VEC1(10),VEC2(10),VEC3(10),V1(10),V2(10)
      DIMENSION PAR1(10),PAR2(10),G1(10),G2(10),Q1(10),Q2(10)
      DIMENSION R11(10,10),R12(10,10),R22(10,10),R12T(10,10)
      dimension parnew1(100),parnew2(100)
C
C     THE PERFORMANCE MEASURE IS THE SUM OVER TIME OF gammaY1 * YC1**2'S,
C     gammaY2 * YC2**2'S,
C     W1*UC**2, AND W2*(UC - U(1))**2.
C
C     THE LQ CONTROLLER GAIN MATRICES ARE G1 AND G2, AND ALPHA IS THE
C     SCALAR GAIN ASSOCIATED WITH THE PAST CONTROL ACTION:
C
C       UNEXT = ALPHA*UC - (G1,XOBS1) - (G2,XOBS2)
C
C     THE SYSTEMS A1,B1,C AND A2,B2,C ARE IN STANDARD OBSERVER FORM, GIVING
C     SPECIAL PROPERTIES TO A1, A2, AND C.
C
C     THE RICCATI UPDATE GIVES V1(I), V2(I), BETA, G1(I), G2(I), ALPHA,
C     R11(I,J), R12(I,J), R12T(I,J), R22(I,J), S, Q1(I), AND Q2(I).
C
C     INITIAL VALUES NEED TO BE ASSIGNED IN THE CALLING PROGRAM:
C        W12 = W1 + W2
C        Q1(I) = 0.0
C        Q2(I) = 0.0
C        S = W1*W2/W12
C        R11(1,1) = gammaY1, WHILE ALL OTHER R11(I,J) = 0.0
C        R12(I,J) = 0.0
C        R22(1,1) = gammaY2, WHILE ALL OTHER R22(I,J) = 0.0
C
C     NOTE THAT A1 IS STORED IN THE FIRST HALF OF PARNEW1, AND B1 IS
C     STORED IN THE LAST HALF OF PARNEW1.  ALSO, A2 IN FIRST HALF OF
C     PARNEW2, AND B2 IN LAST HALF OF PARNEW2.
C
C     THE EQUATIONS THAT ARE ITERATED ONCE PER CALL ARE:
C
C     V1 = R11*B1 + R12*B2 + Q1
C
C     V2 = R21*B1 + R22*B2 + Q2
C
C     BETA = W1 + W2 + S + B1'(V1 + Q1) + B2'(V2 + Q2)
C
C     G1 = (1/BETA)*[-SUM(A1(I)V1(I))  V1(1)  ...  V1(N-1)]'
C
C     G2 = (1/BETA)*[-SUM(A2(I)V2(I))  V2(1)  ...  V2(N-1)]'
C
C     ALPHA = W2/BETA
C
C     R11(NEW) = C'C + A1'(R11)A1 - (BETA)(G1)G1'
C
C     R12(NEW) = A1'(R12)A2 - (BETA)(G1)G2'
C
```

```
C  R22(NEW) = C'C + A2'(R22)A2 - (BETA)(G2)G2'
C
C  S = (1 - ALPHA)W2
C
C  Q1 = (W2)G1
C
C  Q2  = (W2)G2
C
C
C TEMPORARY STORAGE USED TO PERFORM THE ABOVE: B1(N), B2(N), VEC1(N),
C                       VEC2(N), VEC3(N), R12T(N,N)
C

C INITIALIZATION
      W12 = W1 + W2
c Take into account differing na & nb
        N = max(na,nb)
        do i = 1,na
          parnew1(i) = par1(i)
          parnew2(i) = par2(i)
        end do
        do i = 1,nb
          parnew1(N+i) = par1(na+i)
          parnew2(N+i) = par2(na+i)
        end do
c If N2 > 0 this is GPC - re-initialize Ricatti stuff EVERYTIME.
c This is a finite horizon controller.
c NL becomes the finite horizon.
c Everything is iterated NL times.
c Control is assumed to be zero after NU steps.
c If N2 < 0 this is LQ - everything stays the same.
c The Riccati is iterated -N2 times.
        if( N2 .gt. 0 ) then
        nloop = N2
        nuhorizon = NU
        DO I=1,N
        Q1(I) = 0.0
        Q2(I) = 0.0
        END DO
        DO I = 1,N
        DO J = 1,N
        R11(I,J) = 0.0
        R12(I,J) = 0.0
        R22(I,J) = 0.0
        END DO
        END DO
        R11(1,1) = gammaY1
        R22(1,1) = gammaY2
```

```
          S = W1*W2/W12
          else
          nloop = -N2
          nuhorizon = -N2
          end if
c         initialization added
          IF (IFLA.EQ.0.0)THEN
           S = W1*W2/W12
           IFLA = 1
          ENDIF
c---------
          do ii = nloop,1,-1
      DO 10 I=1,N
       B1(I) = PARNEW1(N + I)
       B2(I) = PARNEW2(N + I)
   10 CONTINUE
      CALL MULTMV(R11,B1,VEC1,N)
      CALL MULTMV(R12,B2,VEC2,N)
      CALL ADDVV(VEC1,VEC2,VEC3,N)
      CALL ADDVV(VEC3,Q1,V1,N)
      CALL MULTMV(R12T,B1,VEC1,N)
      CALL MULTMV(R22,B2,VEC2,N)
      CALL ADDVV(VEC1,VEC2,VEC3,N)
      CALL ADDVV(VEC3,Q2,V2,N)
      CALL ADDVV(V1,Q1,VEC1,N)
      CALL ADDVV(V2,Q2,VEC2,N)
      CALL DOTPRO(B1,VEC1,VAL1,N)
      CALL DOTPRO(B2,VEC2,VAL2,N)
      BETA = W12 + S + VAL1 + VAL2
C
C FORMING SPECIAL G1 AND G2
C
      VAL1 = PARNEW1(1)*V1(1)
      VAL2 = PARNEW2(1)*V2(1)
      DO 14 I=2,N
       VAL1 = VAL1 + PARNEW1(I)*V1(I)
       VAL2 = VAL2 + PARNEW2(I)*V2(I)
       G1(I) = V1(I-1)/BETA
       G2(I) = V2(I-1)/BETA
   14 CONTINUE
      G1(1) = -VAL1/BETA
      G2(1) = -VAL2/BETA
C
C FORMING ALPHA
C
      ALPHA = W2/BETA
C
C FORMING NEW R11 AND R22
C
      DO 20 I=1,N
       VEC1(I) = PARNEW1(1)*R11(1,I)
       VEC2(I) = PARNEW2(1)*R22(1,I)
```

```
      DO 16 J=2,N
       VEC1(I) = VEC1(I) + PARNEW1(J)*R11(J,I)
       VEC2(I) = VEC2(I) + PARNEW2(J)*R22(J,I)
 16   CONTINUE
 20 CONTINUE
    VAL1 = PARNEW1(1)*VEC1(1)
    VAL2 = PARNEW2(1)*VEC2(1)
    DO 24 I=2,N
     VAL1 = VAL1 + PARNEW1(I)*VEC1(I)
     VAL2 = VAL2 + PARNEW2(I)*VEC2(I)
 24 CONTINUE
    DO 30 I=N,2,-1
     DO 28 J=N,2,-1
      R11(I,J) = R11(I-1,J-1)
      R22(I,J) = R22(I-1,J-1)
 28  CONTINUE
 30 CONTINUE
    R11(1,1) = VAL1 + 1.0D0
    R22(1,1) = VAL2 + 1.0D0
    DO 34 I=2,N
     R11(1,I) = -VEC1(I-1)
     R11(I,1) = -VEC1(I-1)
     R22(1,I) = -VEC2(I-1)
     R22(I,1) = -VEC2(I-1)
 34 CONTINUE
c----------
        if( ii .le. nuhorizon ) then
    DO 40 I=1,N
     R11(I,I) = R11(I,I) - BETA*G1(I)**2
     R22(I,I) = R22(I,I) - BETA*G2(I)**2
     K = I+1
     DO 38 J=K,N
      R11(I,J) = R11(I,J) - BETA*G1(I)*G1(J)
      R22(I,J) = R22(I,J) - BETA*G2(I)*G2(J)
      R11(J,I) = R11(I,J)
      R22(J,I) = R22(I,J)
 38  CONTINUE
 40 CONTINUE
        end if
c----------
C
C FORMING NEW R12
C
    DO 50 I=1,N
     VEC1(I) = PARNEW1(1)*R12(1,I)
     VEC2(I) = PARNEW2(1)*R12(I,1)
     DO 46 J=2,N
      VEC1(I) = VEC1(I) + PARNEW1(J)*R12(J,I)
      VEC2(I) = VEC2(I) + PARNEW2(J)*R12(I,J)
 46  CONTINUE
 50 CONTINUE
    VAL = PARNEW1(1)*VEC2(1)
```

```
    DO 54 I=2,N
      VAL = VAL + PARNEW1(I)*VEC2(I)
 54 CONTINUE
    DO 60 I=N,2,-1
     DO 58 J=N,2,-1
      R12(I,J) = R12(I-1,J-1)
 58   CONTINUE
 60 CONTINUE
    R12(1,1) = VAL
    DO 64 I=2,N
     R12(1,I) = -VEC1(I-1)
     R12(I,1) = -VEC2(I-1)
 64 CONTINUE
c----------
        if( ii .le. nuhorizon ) then
    DO 70 I=1,N
     DO 68 J=1,N
      R12(I,J) = R12(I,J) - BETA*G1(I)*G2(J)
      R12T(J,I) = R12(I,J)
 68   CONTINUE
 70 CONTINUE
        end if
c----------
C
C FINISHING THE S, Q1, Q2 UPDATES
C
    S = (1.0D0 - ALPHA)*W2
    CALL MULTSV(W2,G1,Q1,N)
    CALL MULTSV(W2,G2,Q2,N)
        end do
c----------
C
    RETURN
    END
C
C
C*************************************
C*******************************
C
C  THIS SUBROUTINE RECREATES THE STATES OF THE SYSTEM
C
C  INPUTS:
C      B1ASSUMED INPUT MATRIX FOR OUTPUT NUMBER 1.
C      B2 //    //  //  //  //    // 2.
C      A1ASSUMED SYSTEM POLYNOMIAL FOR OUTPUT NUMBER 1.
C      A2 //   //      //   //  //   // 2.
C      H1OBSERVER POLYNOMIAL FOR OUTPUT NUMBER 1.
C      H2 //      //  //  //   // 2.
C      XHAT1OBSERVER STATES FOR OUTPUT NUMBER 1.
C      XHAT2 //    // // //    // 2.
C      UPREVIOUS CONTROL ACTION.
C      Y1PLANT OUTPUT NUMBER 1.
```

```
C       Y2 //   // NUMBER 2.
C       NORDER OF THE SYSTEM.
C       TTIME.
C
C   OUTPUT:
C       XHAT1UPDATED VERSION OF OBSERVER STATES FOR OUTPUT NUMBER 1.
C       XHAT2 //   //  //  //   //  //  //  // 2.
C
C
C
C*******************************
      SUBROUTINE OBSERV(B1,B2,A1,A2,C1,C2,error1,error2,H1,H2,
     & XHAT1,XHAT2,U,Y1,Y2,N,T)
      IMPLICIT NONE
      REAL*8 B1(10),B2(10),A1(10),A2(10),C1(10),C2(10),H1(10),H2(10)
      REAL*8XHAT1(10),XHAT2(10),GZERO1(10),GZERO2(10)
      REAL*8U,Y1,Y2,T,YOBS1,YOBS2,ERR1,ERR2,error1,error2
      INTEGERI,N
C******************************
C
C   GENERATE GAINS FOR OBSERVER
C
C*******************************
      DO I=1,N
       GZERO1(I) = A1(I) + H1(I)
       GZERO2(I) = A2(I) + H2(I)
      END DO
C   FIND OBSERVER OUTPUT (YOBS)
      YOBS1 = XHAT1(1)
      YOBS2 = XHAT2(1)
C   FIND OBSERVER ERROR
      ERR1 = Y1 - YOBS1
      ERR2 = Y2 - YOBS2
C   GENERATE ESTIMATE OF X1 AND X2.
      DO I=1,N-1
     XHAT1(I) = A1(I)*YOBS1+XHAT1(I+1)+B1(I)*U
     & +GZERO1(I)*ERR1
     XHAT2(I) = A2(I)*YOBS2+XHAT2(I+1)+B2(I)*U
     &+GZERO2(I)*ERR2
      END DO
      XHAT1(N) = A1(N)*YOBS1 + B1(N)*U + GZERO1(N)*ERR1
      XHAT2(N) = A2(N)*YOBS2 + B2(N)*U+ GZERO2(N)*ERR2
c       WRITE(15,600) T,Y1,YOBS1,ERR1,U
C       WRITE(*,600) T,Y2,YOBS2,ERR2,U
 600  format(7(1X,1Pe9.2))
      RETURN
      END
```

```
c*****************************************************************************
c
c     This subroutine updates the Riccati matrix.  The Riccati matrix is a
c     symmetric matrix.  Also note that, to update this matrix we do not
c     need to recalculate all of the terms in [R].
c
c     The equation used to updat the reccati matrix is as follows;
c        _                                       _   _      _
c       | T            T . T             T|  |    .    |
c       | A1*R11*A1 - B*L1*L1  . A1*R12*A2 - B*L1*L2 | |R11 . R12|
c     [R]= C+|..........................................|=|..........|
c       | T            T . T             T|  |    .    |
c       | A2*R21*A1 - B*L2*L1  . A2*R22*A2 - B*L2*L2 | |R21 . R22|
c       |_                     .                     _| |_   .   _|
c
c
c      NOTE : [R] is 2nx2n matrix.  Therefore R11, R12, R21 and R22 are
c         nxn matrices.
c     Also this subroutine must be called three times to update
c     R11, R12 and R22.  It should be mentioned that R12 = R21^T.
c
c
c
c     INPUT :
c     r2  : nxn matrix which contains the R?? matrix.
c     a1  : nx1 vector of first a coefficients.
c     a2  : nx1 vector of second a coefficients.
c     l1  : nx1 vector of first Riccati gain vector.
c     l2  : nx1 vector of second Riccati gain vector.
c     n   : Dimension of the above matrices.
c     symmetric : 0 => R?? or r2 is not symmetric.
c        1 => R?? or r2 is symmetric.
c
c      NOTE : If R?? is symmetric then l1=l2 and
c     a1=a2.
c
c
c     OUTPUT :
c     r1: Updated version of R??.
c
c
c
c     Written By : Iraj Sadighi
c     Date   : 4-24-1988
c*****************************************************************************

      subroutine rmatrix(r1,r2,a1,a2,l1,l2,b,n,symmetric)
      implicit none
      real*8r1(10,10),r2(10,10),a1(10),a2(10),l1(10),l2(10).b
      integersymmetric.i,j,n
```

```
      if ((symmetric .lt. 0) .and. (symmetric .gt. 1)) then
c     write(*,*) 'ERROR in RMATRIX subroutine!!!!'
c     write(*,*) 'The variable SYMMETRIC should be 0 or 1.'
c     write(*,*) 'Returning to the calling program...'
      return
      end if
c_______________________________________________________ R?? is symmetric.
      if (symmetric .eq. 1) then
      do 10 i=2, n
      do 20 j=i, n
      r1(i,j) = r2((i-1),(j-1)) - b*l1(i)*l2(j)
20    continue
10    continue
      r1(1,1) = 0.0
      do 30 i=2, n
      r1(1,i) = 0.0
      do 40 j=1, n
      r1(1,i) = r1(1,i) + a1(j)*r2((i-1),j)
40    continue
      r1(1,1) = r1(1,1) + r1(1,i)*a1(i-1)
      r1(1,i) = r1(1,i) - b*l1(1)*l2(i)
30    continue
      do 11 i=1, n
      r1(1,1) = r1(1,1) + a1(n)*a1(i)*r2(i,n)
11    continue
      r1(1,1) = r1(1,1) - b*l1(1)*l1(1) + 1.0
      do 50 i=2, n
      do 60 j=1, (i-1)
      r1(i,j) = r1(j,i)
60    continue
50    continue
      return
      end if
c_______________________________________________________ R?? is not symmetric.
      if (symmetric .eq. 0) then
      do 100 i=2, n
      do 70 j=2, n
      r1(i,j) = r2((i-1),(j-1)) - b*l1(i)*l2(j)
70    continue
100   continue
      r1(1,1) = 0.0
      do 80 i=2, n
      r1(1,i) = 0.0
      r1(i,1) = 0.0
      do 90 j=1, n
      r1(1,i) = r1(1,i) + a1(j)*r2(j,(i-1))
      r1(i,1) = r1(i,1) + a2(j)*r2((i-1),j)
90    continue
      r1(1,1) = r1(1,1) + r1(i,1)*a1(i-1)
```

```
      r1(1,i) = r1(1,i) - b*l1(1)*l2(i)
      r1(i,1) = r1(i,1) - b*l1(i)*l2(1)
80    continue

      do 81 i=1, n
      r1(1,1) = r1(1,1) + a1(n)*a2(i)*r2(n,i)
81    continue
      r1(1,1) = r1(1,1) - b*l1(1)*l2(1)
      return
      end if
      return
      end
```

APPENDIX C

Sample Codes for the Simulations in Chapter 4

HOOK.FOR

```
C       EXAMPLE : SVC IS ADDED TO ...
C     This routine executes an interface to ETMSP - providing
C     "hooks" for to input signals into system and retrieve outputs.
C     The purpose is to allow simulation with nonconventional
C     or digital controllers. It also allows access to signals
C     during the run for the purposes of signal analysis, controls
C     or special outputs. Developed at Montana State University
C     in cooperation with the Bonneville Power Administration.
      SUBROUTINE HOOK

      INCLUDE 'PARX.INS/LIST'
      INCLUDE 'CNB8.INS/LIST'
      INCLUDE 'CNYVAR.INS/LIST'
      INCLUDE 'CNB4.INS/LIST'
      INCLUDE 'CNUDM.INS/LIST'
      INCLUDE 'CNVOLT.INS/LIST'
      INCLUDE 'CNMYVA.INS/LIST'
      INCLUDE 'CNNNEW.INS/LIST'
      INCLUDE 'CNSVS.INS/LIST'
      INCLUDE 'CNGEAR.INS/LIST'
      INCLUDE 'CNIPTR.INS/LIST'
      INCLUDE 'CNXSTA.INS/LIST'
      INCLUDE 'CNMSTA.INS/LIST'
      INCLUDE 'CNMDAT.INS/LIST'
      INCLUDE 'CNB5.INS/LIST'
      INCLUDE 'CNLFDA.INS/LIST'
      REAL*4 CHEKC,DCON,TSPAST
      CHARACTER*12 BSNM(20),BSNM2
      INTEGER I,NT,BNUM(20), PT(20),POUT,PRNT,PNOW,II
      C ----------------------------------------------------------------
      C -------------------- START OF USER BLOCK 1 -----------------------
      C             - USER VARIABLE DEFINITIONS -
      REAL*4  REFORIG , VVOLTLO ,VVOLTL,VVOLTLRO,VVOLTLIO,ANGLELO
      REAL*4  VVOLTLON,VVOLTLN,VVOLTLRON,VVOLTLION,ANGLELON
      REAL*4 VVOLTLR,VVOLTLI,ANGLEL,DIFANGLE,VVOLTI2O,VVOLTR2O
      REAL*4 VVOLTLRN,VVOLTLIN,ANGLELN,VVOLTI2ON,VVOLTR2ON
      REAL*4 VVOLT2O,ANGLE2O,VVOLTR2,VVOLTIO,ANGLE2,ANGLEO,VVOLTI2N
      REAL*4 VVOLT2ON,ANGLE2ON,VVOLTR2N,VVOLTION,ANGLE2N,ANGLEON
      REAL*4 XLAMBDA,U,Y1,Y2,PDIAG,QUAL,THETA1(100),THETA2(100),TRACE
      REAL*4 UNEXT,CSIG,REPOWO,REPOW,SSPPEDO,SSPPED,ERROR1,ERROR2
      REAL*4  REPOWON,REPOWN,SSPPEDON,SSPPEDN,DIFFANG,NOIS,RNOIS
      REAL*4  ORIGSVCT,NOISO,Y3,Y4
      INTEGER  IREFPT , SVCNUM ,REFL,ISTART,NA,NBB,IGO,FLAG,IMACC,REFL1
      INTEGER IREFPT2,IREFPT3,IMACCN,IREFPT2N,IREFPT3N,CCC,I1,I2
      INTEGER SVCNUMT,IREFPTT
      DATA I1,I2 /9587,7941/
      COMMON /HKSTUFF/RNOIS
```

```
      CHARACTER*132 CARD
C --------------------- END OF USER BLOCK 1 -------------------------
C -----------------------------------------------------------------
C ********************** INITIALIALIZATION BLOCK ***********************
C              OCCURS WHEN TS = H = 0.0
        IF ((TS .EQ. 0.0) .AND. (H .EQ. 0.0)) THEN
        OPEN(UNIT=21, FILE='HK.DAT',STATUS='NEW')
      C -----------------------------------------------------------------
      C -------------------- START OF USER BLOCK 2 ------------------------
      C         - ASSIGN NT, POUT AND EACH BSNM(I) UP TO I = NT. -
      OPEN(UNIT=22, FILE='HK2.DAT',STATUS='NEW')
      OPEN(UNIT=24, FILE='HK3.DAT',STATUS='NEW')
      OPEN(UNIT=23, FILE='THETA1.DAT',STATUS='NEW')
           POUT = 5
      NT = 4
      CGET NAMES FOR BUSES
       BSNM(1) = 'KEELER   230'
      BSNM(2) = 'MAPLE VL 230'
       BSNM(3) = 'CHIEF JO13.8'
      cBSNM(4) = 'ROCKY RH 115'
      cBSNM(4) = 'ARAPAHOE115.'
      BSNM(4) = 'LARSON   115'
C --------------------- END OF USER BLOCK 2 -------------------------
C -----------------------------------------------------------------
      PRNT = POUT
      TSPAST = -1.0
C     GET NUMBERS FOR BUSES
      DO 10 I=1,NT
         CALL CHKBUS(BSNM(I),BNUM(I))

C     CHECK FOR BUS # = 0, WHICH INDICATES A PROBLEM.
C     IF OK, THEN GET GENERATOR NUMBERS IN INTERNAL ORDER.
         IF( BNUM(I) .NE. 0) THEN
          PT(I) = NNEW(BNUM(I))
         ELSE
          WRITE(6,*) ' ERROR - PROBLEM WITH BUS NAME IN HOOK '
          CALL ASTOP('HOOK ERROR',16)
         ENDIF
10     CONTINUE
C -----------------------------------------------------------------
C -------------------- START OF USER BLOCK 3 -----------------------
C  - GET THE INDICES OF ANY RELEVENT STATES USING PT(I) AS BUS INDEX -

C     THE NUMBER OF THE SVC IN DYNAMIC DATA FILE
      CCC=1
      SVCNUM = 1
      SVCNUMT= 2
C     GET THE SVC Y-VARIABLE FIELD OFFSET
      IREFPT = JVSVR(SVCNUM)
      IREFPTT=JVSVR(SVCNUMT)
C     GET THE SVC REFERENCE VALUE
c     REFORIG = BSVC(IREFPT + 2)
```

```
      ORIGSVCT = BSVC(IREFPTT + 2)
      REFORIG = RVREF(IREFPT + 1)
C     GET SVC BUS NUMBER(NEW)
C THIS GIVES BUS NUMBER
     REFL = NROLD(SVCNUM)
C THIS GIVE MACHINE NUMBER
C        REFL = NTHG(PT(2))
C     GET SVC BUS INITIAL VOLTAGE
      VVOLTLO = ABS(VOLT(REFL))
C     GET STATIONN BUS VOLTAGE
C     IMACC = NTHG(PT(1))
C BUS NUMBER IS USED FOR THE KEELER
      IMACC = PT(1)
      IMACCN=PT(3)
      IMACCN=NTHG(PT(3))
   IREFPT2=MYVAR(1,PT(1))
      IREFPT2N=MYVAR(1,PT(3))
      REPOWO=PDG(IREFPT2+2)
      REPOWON=PDG(IREFPT2N+2)
      IREFPT3=MSTATE(1,PT(1))
      IREFPT3N=MSTATE(1,PT(3))
      SSPPEDO=XSTATE(IREFPT3+1)
      SSPPEDON=XSTATE(IREFPT3N+1)
      VVOLT2O = ABS(VOLT(IMACC))
      VVOLTR2O = REAL(VOLT(IMACC))
      VVOLTI2O = AIMAG(VOLT(IMACC))
      VVOLT2ON = ABS(VOLT(IMACCN))
      VVOLTR2ON = REAL(VOLT(IMACCN))
      VVOLTI2ON = AIMAG(VOLT(IMACCN))
C     GET THE DIFFERENCE BETWEEN THE PHASE ANGLE OF THE BUSES
      ANGLE2O = (180.0/3.14)*ATAN2(VVOLTI2O,VVOLTR2O)
      ANGLE2ON = (180.0/3.14)*ATAN2(VVOLTI2ON,VVOLTR2ON)
      VVOLTLRO = REAL(VOLT(REFL))
      VVOLTLIO = AIMAG(VOLT(REFL))
C     ANGLELO =(180.0/3.14)*ATAN2(VVOLTLIO,VVOLTLRO)-ANGLE2O
      ANGLELO=ANGLE2ON-ANGLE2O
      ANGLEO = ANGLELO
C        - MAKE ANY INITIAL WRITES TO UNIT 21 -
      WRITE(21,1000)
C     WRITE(21,1001)
C     WRITE(21,1002)
1000  FORMAT('DATA WITH SVC ON MAPLE VL 230. ')
1001  FORMAT(' ')
1002  FORMAT('3')
C --------------------- END OF USER BLOCK 3 ------------------------
C ---------------------------------------------------------------------
C ************** END OF INITIALIZATION BLOCK **********************
      ENDIF
```

```
C      CHECK FOR REPETE CALL AT SAME TIME BECAUSE OF DISCONTINUITY.
C      IF DCON = 1 THEN SET CONTROL ACTION FOR THIS CALL TO WHAT IT WAS
C      FOR PREVIOUS CALL OR JUST LET IT BE RECALCULATED.

       IF( TS .EQ. TSPAST) THEN
         DCON = 1.0
       ELSE
         DCON = 0.0
       ENDIF
       TSPAST = TS
C      END OF DISCONTINUITY CHECK
C      TEST FOR PRINTING
       IF (PRNT.EQ.POUT) THEN
         PNOW = 1
         PRNT = 1
       ELSEIF (PRNT.NE.POUT) THEN
         PNOW = 0
         PRNT = PRNT + 1
       ENDIF
       IF (DCON.EQ.1) THEN
         BACKSPACE(UNIT=21)
         PNOW = 1
         PRNT = PRNT - 1
       ENDIF
C -------------------------------------------------------------------
C -------------------- START OF USER BLOCK 4 ----------------------
C                 - CONTROL ACTIONS -
C      CHANGE SVC REFERENCE VALUE
C      IF ((TS.GE.0.0).AND.(TS.LT.0.5)) THEN
C       RVREF(IREFPT + 1) = REFORIG
C      ELSE
C        RVREF(IREFPT + 1) = REFORIG
C      ENDIF
C
C    NO FEEDBACK
       RVREF(IREFPT+1) = REFORIG + UNEXT
c      BSVC(IREFPT+2) = REFORIG-UNEXT
C      IF ((TS.GE.4.0).AND.(TS.LE.5.0)) THEN
C      BSVC(IREFPT+2) = REFORIG + UNEXT + 0.0
C      ENDIF
       VVOLTL = VVOLTLO - ABS(VOLT(REFL))
       VVOLTR2 = REAL(VOLT(IMACC))
       VVOLTI2 = AIMAG(VOLT(IMACC))
       ANGLE2 = (180.0/3.14)*ATAN2(VVOLTI2,VVOLTR2)
       VVOLTR2N = REAL(VOLT(IMACCN))
       VVOLTI2N = AIMAG(VOLT(IMACCN))
       ANGLE2N = (180.0/3.14)*ATAN2(VVOLTI2N,VVOLTR2N)
       VVOLTLR = REAL(VOLT(REFL))
       VVOLTLI = AIMAG(VOLT(REFL))
C      ANGLEL =(180.0/3.14)*ATAN2(VVOLTLI,VVOLTLR)-ANGLE2
       ANGLEL =ANGLE2N-ANGLE2
       DIFANGLE = ANGLELO - ANGLEL
```

```
      DIFFANG=(DIFANGLE-ANGLEO)/0.01
      ANGLEO = DIFANGLE
      REPOW=PDG(IREFPT2+2)
      SSPPED=XSTATE(IREFPT3+1)
      REPOWN=PDG(IREFPT2N+2)
      SSPPEDN=XSTATE(IREFPT3N+1)
C     USE IDENTIFIER
      NA = 5
      NBB = NA
      NC=3
      N1=NA+NBB+NC
      XLAMBDA = 0.97
      PDIAG =1.0D6
      ISTART = 1
      IF (FLAG.EQ.0)THEN
        ISTART = 0
        FLAG = 1
      ENDIF
      CSIG = UNEXT
      U = CSIG
C     Y1 = XSTATE(IREFPT3N+1)
      Y1 = DIFANGLE
      Y2 =0
C     Y2= VVOLTL
      IF (CCC.EQ.5) THEN
   CALL UUD1(U,Y1,XLAMBDA,N1,NA,NBB,ISTART,QUAL,THETA1,TRACE
  & ,PDIAG,ERROR1,y3)
   CALL UUD2(U,Y2,XLAMBDA,N1,NA,NBB,ISTART,QUAL,THETA2,TRACE
  & ,PDIAG,ERROR2,y4)
      IGO = 2
      IF (TS .LE. 5) THEN
      IGO = 1
      ENDIF
      IF (TS .LE. 1) THEN
      IGO = 0
      ENDIF
      CALL CONTROL(THETA1,THETA2,Y1,Y2,CSIG,NA,QUAL,IGO,TS
  &   ,ERROR1,ERROR2,UNEXT)
      CCC=1
      ENDIF
      CCC=CCC+1
C
C     IF (TS.GE.5.0) THEN
C
C     write(card,5000) 'SLD',' ',pt(4),0,noiso,0.0
C5000format(A3,A1,2I5,8F10.5)
C     call moload(card,1)
C
      NOIS= 0.0
      DO II=1,12
        NOIS = NOIS + RAN(I1,I2)
      END DO
C
```

```
C Reset mean of distribution
      NOIS = NOIS - 6.0
      NOIS=NOIS*3
C     NOIS=0
      RNOIS=NOIS
C     nois =((-1)**ccc)*4
C     BSVC(IREFPTT+2)=ORIGSVCT+NOIS
c     nois =0.0
C     write(card,5000) 'SLD',' ',pt(4),0,nois,0.0
C5000format(A3,A1,2I5,8F10.5)
C     call moload(card,1)
C     NOISO=NOIS
C     ENDIF
C --------------------- END OF USER BLOCK 4 ------------------------
C -----------------------------------------------------------------
      IF (PNOW) THEN
C -----------------------------------------------------------------
C -------------------- START OF USER BLOCK 5 -----------------------
C                 - WRITE TO UNIT 21 -
      WRITE(21,2000)TS,NOIS,BSVC(IREFPT+2),DIFANGLE
2000  FORMAT(1X,F7.4,2X,3(F10.5,2X))
      WRITE(22,3000) TS,ABS(VOLT(REFL)),PDG(IREFPT2N+2)
   &      ,SSPPEDN
      WRITE(24,3000) TS,ABS(VOLT(PT(2)))
      WRITE(23,4000) TS,(THETA1(I), I=1,10)
3000  FORMAT(1X,F7.4,1X,3(F10.5,1X))
4000  FORMAT(1X,F7.4,1X,10(F10.5,1X))
C --------------------- END OF USER BLOCK 5 ------------------------
C -----------------------------------------------------------------
      ENDIF
      RETURN
      END
```

CONTROL.FOR

```
C***************************************
C  THIS SUBROUTINE MAKES A DECISION ON THE TYPE OF CONTROL
C  ACTION AND THEN CREATES THE CONTROL ACTION.
C
C  INPUTS:
C     THETA1CHARACTERISTIC POLYNOMIAL OF THE PLANT, FOR OUTPUT NUMBER 1.
C     THETA2   //      //    //// //  //  //  // 2.
C     B1CONTROL INPUT VECTOR OF THE PLANT, FOR OUTPUT NUMBER 1.
C     B2 //  //  // // // // // //  // 2.
C     Y1PLANT OUTPUT NUMBER 1.
C     Y2PLANT OUTPUT NUMBER 2.
C     UPLANT INPUT.
C     NSYSTEM ORDER.
C     QUALQUALITY OF A AND B COEFFICIENTS.(0-1)
C     IGOFLAG FOR ADAPTIVE CONTROL.
C     TTIME.
C
C  OUTPUT:
C     UNEXTCONTROL ACTION.
C
C  Written By : D. Trudnowski
C  Changed By : F. Fatehi
C***************************************

      SUBROUTINE CONTROL(THETA1,THETA2,Y1,Y2,U,N,QUAL,IGO,T
     &  ,ERROR1,ERROR2,UNEXT)
      IMPLICIT NONE
      REAL*4 THETA1(1),THETA2(1)
      REAL*4 A1(10),A2(10),B1(10),B2(10),C1(10),C2(10),L1(10),L2(10)
      REAL*4  R11(10,10), R12(10,10), R21(10,10), R22(10,10)
      REAL*4 TR11(10,10),TR12(10,10),TR21(10,10),TR22(10,10)
      REAL*4 XHAT1(10),XHAT2(10),H1(10),H2(10),GZERO1(10),GZERO2(10)
      REAL*4 ALPHA,U,Y1,Y2,QUAL,T,UNEXT,UPROB,FIRST,ERROR1,ERROR2
      REAL*4 W1,W2,W12,G1(10),G2(10),gammaY1,gammaY2,PAR1(10),PAR2(10)
      INTEGER PFLG,I,J,ICOUNT,IGO,N,N2,NU,NA,NB,NM

C*******************************
C  FIRST TIME THROUGH INITIALIZE MATRICES
C*******************************
        IF (ICOUNT.GE.80) ICOUNT=0
      IF (FIRST.EQ.0.0) THEN
       PFLG = 0
       DO I=1,N
        DO J=1,N
         R11(I,J) = 0.0
           R12(I,J) = 0.0
           R21(I,J) = 0.0
           R22(I,J) = 0.0
         TR11(I,J) = 0.0
```

```
          TR12(I,J) = 0.0
          TR21(I,J) = 0.0
          TR22(I,J) = 0.0
       END DO
         A1(I)   = 0.0
         A2(I)   = 0.0
         B1(I)   = 0.0
         B2(I)   = 0.0
       C1(I)   = 0.0
         C2(I)   = 0.0
       H1(I)   = 0.0
         H2(I)   = 0.0
       G1(I)   = 0.0
         G2(I)   = 0.0
         XHAT1(I) = 0.0
         XHAT2(I) = 0.0
         GZERO1(I) = 0.0
         GZERO2(I) = 0.0
      END DO
C ADDED INITIALIZATION
        W1 = 2.5
        W2 = .1
        W12 = W1 + W2
        N2 = -1
        NU = 5
        gammaY1 = 1
        gammaY2 = 1

C      PAR1 = THETA1
C       PAR2 = THETA2
        NA = N
        NB = N
        R11(1,1) = 1.0
      R22(1,1) = 1.0
        ALPHA  = 0.0
      FIRST  = 1.0
      ICOUNT = 0
     END IF
C********************************
C
C      UPDATE A1, A2, B1 AND B2 VECTORS.
C
C********************************
     DO I=1,N
      A1(I) = -THETA1(I)
      B1(I) =  THETA1(N+I)
      C1(I) =  THETA1(2*N+I)
      A2(I) = -THETA2(I)
      B2(I) =  THETA2(N+I)
      C2(I) =  THETA2(2*N+I)
     ENDDO
```

```
C*******************************
C  IF THE FLAG IGO IS 0 THEN THE CONTROL ACTION IS NOT
C  DETERMINED BY THE ADAPTIVE CONTROL
C********************************
      UPROB = 0.05
      IF (IGO.EQ.0) THEN

       UNEXT =  UPROB
       IF (ICOUNT .LE. 73) UNEXT = -UPROB
       IF (ICOUNT .LE. 65) UNEXT =  UPROB
       IF (ICOUNT .LE. 61) UNEXT = -UPROB
       IF (ICOUNT .LE. 54) UNEXT =  UPROB
       IF (ICOUNT .LE. 46) UNEXT = -UPROB
       IF (ICOUNT .LE. 42) UNEXT =  UPROB
       IF (ICOUNT .LE. 34) UNEXT = -UPROB
       IF (ICOUNT .LE. 27) UNEXT =  UPROB
       IF (ICOUNT .LE. 23) UNEXT = -UPROB
       IF (ICOUNT .LE. 16) UNEXT =  UPROB
       IF (ICOUNT .LE. 13) UNEXT = -UPROB
       IF (ICOUNT .LE. 5)  UNEXT =  UPROB
       ICOUNT = ICOUNT +1
       RETURN
      ENDIF
      IF (IGO .GE. 1) THEN
C********************************
C  START THE ADAPTIVE CONTROL
C********************************
      CALL RICTWO(THETA1,THETA2,W1,W2,W12,G1,G2,ALPHA,
     &      R11,R12,R22,NA,NB,N2,NU,
     &    gammaY1,gammaY2)
      CALL OBSERV(B1,B2,A1,A2,C1,C2,ERROR1,ERROR2
     &   ,H1,H2,XHAT1,XHAT2,U,Y1,Y2,N,T)
       IF (IGO .EQ. 1) THEN

       UNEXT =  UPROB
       IF (ICOUNT .LE. 73) UNEXT = -UPROB
       IF (ICOUNT .LE. 65) UNEXT =  UPROB
       IF (ICOUNT .LE. 61) UNEXT = -UPROB
       IF (ICOUNT .LE. 54) UNEXT =  UPROB
       IF (ICOUNT .LE. 46) UNEXT = -UPROB
       IF (ICOUNT .LE. 42) UNEXT =  UPROB
       IF (ICOUNT .LE. 34) UNEXT = -UPROB
       IF (ICOUNT .LE. 27) UNEXT =  UPROB
       IF (ICOUNT .LE. 23) UNEXT = -UPROB
       IF (ICOUNT .LE. 16) UNEXT =  UPROB
       IF (ICOUNT .LE. 13) UNEXT = -UPROB
       IF (ICOUNT .LE. 5)  UNEXT =  UPROB
       ICOUNT = ICOUNT + 1
       IF(ICOUNT .GT. 80) THEN
        ICOUNT = 0
        UPROB = ABS(UPROB) + 0.08
       ENDIF
```

```
      WRITE(16,601) T,(G1(J),J=1,N),ALPHA
C       WRITE(*,601) T,(G2(J),J=1,N),ALPHA
 601    format(8(1X,1Pe10.3))
      RETURN
      ENDIF
     ENDIF
C*******************************
C  CALCULATE THE CONTROL SIGNAL
C********************************
     IF (IGO .EQ. 2) THEN
      UNEXT = ALPHA * U
      DO J=1,N
        UNEXT = UNEXT - G1(J)*XHAT1(J) - G2(J)*XHAT2(J)
      END DO
     ENDIF
     WRITE(16,602) T,(G1(J),J=1,N),ALPHA
C     WRITE(*,602) T,(G2(J),J=1,N),ALPHA
602  format(8(1X,1Pe10.3))
C
      IF (UNEXT.GT.0.8) THEN
        UNEXT = 0.8
      ELSE IF (UNEXT.LT.-0.8) THEN
        UNEXT = -0.8
      END IF

     RETURN
     END
C
C
     SUBROUTINE RICTWO(PAR1,PAR2,W1,W2,W12,G1,G2,
    &            ALPHA,
    &            R11,R12,R22,NA,NB,
    &            N2,NU,gammaY1,gammaY2)
     IMPLICIT REAL*4 (A-H,O-Z)
     DIMENSION B1(10),B2(10),VEC1(10),VEC2(10),VEC3(10),V1(10),V2(10)
     DIMENSION PAR1(10),PAR2(10),G1(10),G2(10),Q1(10),Q2(10)
     DIMENSION R11(10,10),R12(10,10),R22(10,10),R12T(10,10)
     dimension parnew1(100),parnew2(100)
C
C  THE PERFORMANCE MEASURE IS THE SUM OVER TIME OF gammaY1 * YC1**2'S,
C  gammaY2 * YC2**2'S,
C  W1*UC**2, AND W2*(UC - U(1))**2.
C
C  THE LQ CONTROLLER GAIN MATRICES ARE G1 AND G2, AND ALPHA IS THE
C  SCALAR GAIN ASSOCIATED WITH THE PAST CONTROL ACTION:
C
C    UNEXT = ALPHA*UC - (G1,XOBS1) - (G2,XOBS2)
C
C  THE SYSTEMS A1,B1,C AND A2,B2,C ARE IN STANDARD OBSERVER FORM, GIVING
C  SPECIAL PROPERTIES TO A1, A2, AND C.
C
C  THE RICCATI UPDATE GIVES V1(I), V2(I), BETA, G1(I), G2(I), ALPHA,
```

```
C  R11(I,J), R12(I,J), R12T(I,J), R22(I,J), S, Q1(I), AND Q2(I).
C
C  INITIAL VALUES NEED TO BE ASSIGNED IN THE CALLING PROGRAM:
C     W12 = W1 + W2
C     Q1(I) = 0.0
C     Q2(I) = 0.0
C     S = W1*W2/W12
C     R11(1,1) = gammaY1, WHILE ALL OTHER R11(I,J) = 0.0
C     R12(I,J) = 0.0
C     R22(1,1) = gammaY2, WHILE ALL OTHER R22(I,J) = 0.0
C
C  NOTE THAT A1 IS STORED IN THE FIRST HALF OF PARNEW1, AND B1 IS
C  STORED IN THE LAST HALF OF PARNEW1.  ALSO, A2 IN FIRST HALF OF
C  PARNEW2, AND B2 IN LAST HALF OF PARNEW2.
C
C  THE EQUATIONS THAT ARE ITERATED ONCE PER CALL ARE:
C
C  V1 = R11*B1 + R12*B2 + Q1
C
C  V2 = R21*B1 + R22*B2 + Q2
C
C  BETA = W1 + W2 + S + B1'(V1 + Q1) + B2'(V2 + Q2)
C
C  G1 = (1/BETA)*[-SUM(A1(I)V1(I))  V1(1) ... V1(N-1)]'
C
C  G2 = (1/BETA)*[-SUM(A2(I)V2(I))  V2(1) ... V2(N-1)]'
C
C  ALPHA = W2/BETA
C
C  R11(NEW) = C'C + A1'(R11)A1 - (BETA)(G1)G1'
C
C  R12(NEW) = A1'(R12)A2 - (BETA)(G1)G2'
C
C  R22(NEW) = C'C + A2'(R22)A2 - (BETA)(G2)G2'
C
C  S = (1 - ALPHA)W2
C
C  Q1 = (W2)G1
C
C  Q2  = (W2)G2
C
C
C TEMPORARY STORAGE USED TO PERFORM THE ABOVE: B1(N), B2(N), VEC1(N),
C                   VEC2(N), VEC3(N), R12T(N,N)
C

C INITIALIZATION
      W12 = W1 + W2
```

```
C Take into account differing na & nb
        N = max(na,nb)
        do i = 1,na
           parnew1(i) = par1(i)
           parnew2(i) = par2(i)
        end do
        do i = 1,nb
           parnew1(N+i) = par1(na+i)
           parnew2(N+i) = par2(na+i)
        end do
C If N2 > 0 this is GPC - re-initialize Ricatti stuff EVERYTIME.
C This is a finite horizon controller.
C NL becomes the finite horizon.
c Everything is iterated NL times.
C Control is assumed to be zero after NU steps.
C If N2 < 0 this is LQ - everything stays the same.
C The Riccati is iterated -N2 times.
        if( N2 .gt. 0 ) then
        nloop = N2
        nuhorizon = NU
        DO I=1,N
        Q1(I) = 0.0
        Q2(I) = 0.0
        END DO
        DO I = 1,N
        DO J = 1,N
        R11(I,J) = 0.0
        R12(I,J) = 0.0
        R22(I,J) = 0.0
        END DO
        END DO
        R11(1,1) = gammaY1
        R22(1,1) = gammaY2
        S = W1*W2/W12
        else
        nloop = -N2
        nuhorizon = -N2
        end if
C        initialization added
        IF (IFLA.EQ.0.0)THEN
         S = W1*W2/W12
         IFLA = 1
        ENDIF
C---------
        do ii = nloop,1,-1
     DO 10 I=1,N
      B1(I) = PARNEW1(N + I)
      B2(I) = PARNEW2(N + I)
   10 CONTINUE
     CALL MULTMV(R11,B1,VEC1,N)
     CALL MULTMV(R12,B2,VEC2,N)
```

```
      CALL ADDVV(VEC1,VEC2,VEC3,N)
      CALL ADDVV(VEC3,Q1,V1,N)
      CALL MULTMV(R12T,B1,VEC1,N)
      CALL MULTMV(R22,B2,VEC2,N)
      CALL ADDVV(VEC1,VEC2,VEC3,N)
      CALL ADDVV(VEC3,Q2,V2,N)
      CALL ADDVV(V1,Q1,VEC1,N)
      CALL ADDVV(V2,Q2,VEC2,N)
      CALL DOTPRO(B1,VEC1,VAL1,N)
      CALL DOTPRO(B2,VEC2,VAL2,N)
      BETA = W12 + S + VAL1 + VAL2
C
C FORMING SPECIAL G1 AND G2
C
      VAL1 = PARNEW1(1)*V1(1)
      VAL2 = PARNEW2(1)*V2(1)
      DO 14 I=2,N
        VAL1 = VAL1 + PARNEW1(I)*V1(I)
        VAL2 = VAL2 + PARNEW2(I)*V2(I)
        G1(I) = V1(I-1)/BETA
        G2(I) = V2(I-1)/BETA
   14 CONTINUE
      G1(1) = -VAL1/BETA
      G2(1) = -VAL2/BETA
C
C FORMING ALPHA
C
      ALPHA = W2/BETA
C
C FORMING NEW R11 AND R22
C
      DO 20 I=1,N
        VEC1(I) = PARNEW1(1)*R11(1,I)
        VEC2(I) = PARNEW2(1)*R22(1,I)
        DO 16 J=2,N
          VEC1(I) = VEC1(I) + PARNEW1(J)*R11(J,I)
          VEC2(I) = VEC2(I) + PARNEW2(J)*R22(J,I)
   16   CONTINUE
   20 CONTINUE
      VAL1 = PARNEW1(1)*VEC1(1)
      VAL2 = PARNEW2(1)*VEC2(1)
      DO 24 I=2,N
        VAL1 = VAL1 + PARNEW1(I)*VEC1(I)
        VAL2 = VAL2 + PARNEW2(I)*VEC2(I)
   24 CONTINUE
      DO 30 I=N,2,-1
        DO 28 J=N,2,-1
          R11(I,J) = R11(I-1,J-1)
          R22(I,J) = R22(I-1,J-1)
   28   CONTINUE
   30 CONTINUE
      R11(1,1) = VAL1 + 1.0D0
      R22(1,1) = VAL2 + 1.0D0
```

```
      DO 34 I=2,N
        R11(1,I) = -VEC1(I-1)
        R11(I,1) = -VEC1(I-1)
        R22(1,I) = -VEC2(I-1)
        R22(I,1) = -VEC2(I-1)
   34 CONTINUE
C----------
        if( ii .le. nuhorizon ) then
      DO 40 I=1,N
        R11(I,I) = R11(I,I) - BETA*G1(I)**2
        R22(I,I) = R22(I,I) - BETA*G2(I)**2
        K = I+1
        DO 38 J=K,N
          R11(I,J) = R11(I,J) - BETA*G1(I)*G1(J)
          R22(I,J) = R22(I,J) - BETA*G2(I)*G2(J)
          R11(J,I) = R11(I,J)
          R22(J,I) = R22(I,J)
   38   CONTINUE
   40 CONTINUE
        end if
C----------
C
C  FORMING NEW R12
C
      DO 50 I=1,N
        VEC1(I) = PARNEW1(1)*R12(1,I)
        VEC2(I) = PARNEW2(1)*R12(I,1)
        DO 46 J=2,N
          VEC1(I) = VEC1(I) + PARNEW1(J)*R12(J,I)
          VEC2(I) = VEC2(I) + PARNEW2(J)*R12(I,J)
   46   CONTINUE
   50 CONTINUE
      VAL = PARNEW1(1)*VEC2(1)
      DO 54 I=2,N
        VAL = VAL + PARNEW1(I)*VEC2(I)
   54 CONTINUE
      DO 60 I=N,2,-1
        DO 58 J=N,2,-1
          R12(I,J) = R12(I-1,J-1)
   58   CONTINUE
   60 CONTINUE
      R12(1,1) = VAL
      DO 64 I=2,N
        R12(1,I) = -VEC1(I-1)
        R12(I,1) = -VEC2(I-1)
   64 CONTINUE
C----------
        if( ii .le. nuhorizon ) then
      DO 70 I=1,N
        DO 68 J=1,N
          R12(I,J) = R12(I,J) - BETA*G1(I)*G2(J)
          R12T(J,I) = R12(I,J)
```

```
   68  CONTINUE
   70 CONTINUE
         end if
C----------
C
C  FINISHING THE S, Q1, Q2 UPDATES
C
    S = (1.0D0 - ALPHA)*W2
    CALL MULTSV(W2,G1,Q1,N)
    CALL MULTSV(W2,G2,Q2,N)
         end do
C----------
C
    RETURN
    END
C
C
C***********************************
C******************************
C
C   THIS SUBROUTINE RECREATES THE STATES OF THE SYSTEM
C
C   INPUTS:
C       B1ASSUMED INPUT MATRIX FOR OUTPUT NUMBER 1.
C       B2 //    //  //  //  //   // 2.
C       A1ASSUMED SYSTEM POLYNOMIAL FOR OUTPUT NUMBER 1.
C       A2 //  //      //  //  //   // 2.
C       H1OBSERVER POLYNOMIAL FOR OUTPUT NUMBER 1.
C       H2 //     //  //  //   // 2.
C       XHAT1OBSERVER STATES FOR OUTPUT NUMBER 1.
C       XHAT2 //     // // //    // 2.
C       UPREVIOUS CONTROL ACTION.
C       Y1PLANT OUTPUT NUMBER 1.
C       Y2 //  //  NUMBER 2.
C       NORDER OF THE SYSTEM.
C       TTIME.
C
C   OUTPUT:
C       XHAT1UPDATED VERSION OF OBSERVER STATES FOR OUTPUT NUMBER 1.
C       XHAT2 //  //  //  //    // //  //   // 2.
C
C
C
C*******************************
     SUBROUTINE OBSERV(B1,B2,A1,A2,C1,C2,ERROR1,ERROR2
    & ,H1,H2,XHAT1,XHAT2,U,Y1,Y2,N,T)
     IMPLICIT NONE
     REAL*4 B1(10),B2(10),A1(10),A2(10),C1(10),C2(10),H1(10),H2(10)
     REAL*4XHAT1(10),XHAT2(10),GZERO1(10),GZERO2(10)
     REAL*4U,Y1,Y2,T,YOBS1,YOBS2,ERR1,ERR2,ERROR1,ERROR2
     INTEGERI,N
```

```
C******************************
C
C  GENERATE GAINS FOR OBSERVER
C
C*******************************
      DO I=1,N
       GZERO1(I) = A1(I) + H1(I)
       GZERO2(I) = A2(I) + H2(I)
      END DO
C  FIND OBSERVER OUTPUT (YOBS)
      YOBS1 = XHAT1(1)
      YOBS2 = XHAT2(1)
C  FIND OBSERVER ERROR
      ERR1 = Y1 - YOBS1
      ERR2 = Y2 - YOBS2
C  GENERATE ESTIMATE OF X1 AND X2.
      DO I=1,N-1
       XHAT1(I) = A1(I)*YOBS1+XHAT1(I+1)+B1(I)*U+C1(I)*ERROR1
     &  +GZERO1(I)*ERR1
       XHAT2(I) = A2(I)*YOBS2+XHAT2(I+1)+B2(I)*U+C2(I)*ERROR2
     &  +GZERO2(I)*ERR2
      END DO
      XHAT1(N) = A1(N)*YOBS1 + B1(N)*U +C1(N)*ERROR1
     &  + GZERO1(N)*ERR1
      XHAT2(N) = A2(N)*YOBS2 + B2(N)*U +C2(N)*ERROR2
     & + GZERO2(N)*ERR2
      WRITE(15,600) T,Y1,YOBS1,ERR1,U
C      WRITE(*,600) T,Y2,YOBS2,ERR2,U
600  format(7(1X,1Pe9.2))
      RETURN
      END
```

```
c****************************************************************************
c
c       This subroutine updates the Riccati matrix.  The Riccati matrix is a
c       symmetric matrix.  Also note that, to update this matrix we do not
c       need to recalculate all of the terms in [R].
c
c       The equation used to updat the reccati matrix is as follows;
c
c           _                                 _ _      _
c           | T          T   T                T||   .  |
c           | A1*R11*A1 - B*L1*L1  . A1*R12*A2 - B*L1*L2 ||R11 . R12|
c       [R]= C+|.........................................|=|..........|
```

```
c        | T            T . T          T||  .  |
c        | A2*R21*A1 - B*L2*L1  . A2*R22*A2 - B*L2*L2 ||R21 . R22|
c        |_                .               _||_  .  _|
c
c
c         NOTE : [R] is 2nx2n matrix.  Therefore R11, R12, R21 and R22 are
c            nxn matrices.
c        Also this subroutine must be called three times to update
c        R11, R12 and R22.  It should be mentioned that R12 = R21^T.
c
c
c
c      INPUT :
c      r2  : nxn matrix which contains the R?? matrix.
c      a1  : nx1 vector of first a coefficients.
c      a2  : nx1 vector of second a coefficients.
c      l1  : nx1 vector of first Riccati gain vector.
c      l2  : nx1 vector of second Riccati gain vector.
c      n  : Dimension of the above matrices.
c      symmetric : 0 => R?? or r2 is not symmetric.
c         1 => R?? or r2 is symmetric.
c
c       NOTE : If R?? is symmetric then l1=l2 and
c      .a1=a2.
c
c
c      OUTPUT :
c      r1: Updated version of R??.
c
c
c
c      Written By : Iraj Sadighi
c      Date   : 4-24-1988
c***************************************************************************

      subroutine rmatrix(r1,r2,a1,a2,l1,l2,b,n,symmetric)
      implicit none
      real*4r1(10,10),r2(10,10),a1(10),a2(10);l1(10),l2(10),b
      integersymmetric,i,j,n

      if ((symmetric .lt. 0) .and. (symmetric .gt. 1)) then
      write(*,*) 'ERROR in RMATRIX subroutine!!!!'
      write(*,*) 'The variable SYMMETRIC should be 0 or 1.'
      write(*,*) 'Returning to the calling program...'
      return
      end if
c_______________________________________________________________ R?? is symmetric.
      if (symmetric .eq. 1) then
```

```
      do 10 i=2, n
      do 20 j=i, n
      r1(i,j) = r2((i-1),(j-1)) - b*l1(i)*l2(j)
20    continue
10    continue
      r1(1,1) = 0.0
      do 30 i=2, n
      r1(1,i) = 0.0
      do 40 j=1, n
      r1(1,i) = r1(1,i) + a1(j)*r2((i-1),j)
40    continue
      r1(1,1) = r1(1,1) + r1(1,i)*a1(i-1)
      r1(1,i) = r1(1,i) - b*l1(1)*l2(i)
30    continue
      do 11 i=1, n
      r1(1,1) = r1(1,1) + a1(n)*a1(i)*r2(i,n)
11    continue
      r1(1,1) = r1(1,1) - b*l1(1)*l1(1) + 1.0
      do 50 i=2, n
      do 60 j=1, (i-1)
      r1(i,j) = r1(j,i)
60    continue
50    continue
      return
      end if
c_______________________________________________________ R?? is not symmetric.
      if (symmetric .eq. 0) then
      do 100 i=2, n
      do 70 j=2, n
      r1(i,j) = r2((i-1),(j-1)) - b*l1(i)*l2(j)
70    continue
100   continue
      r1(1,1) = 0.0
      do 80 i=2, n
      r1(1,i) = 0.0
      r1(i,1) = 0.0
      do 90 j=1, n
      r1(1,i) = r1(1,i) + a1(j)*r2(j,(i-1))
      r1(i,1) = r1(i,1) + a2(j)*r2((i-1),j)
90    continue
      r1(1,1) = r1(1,1) + r1(i,1)*a1(i-1)
      r1(1,i) = r1(1,i) - b*l1(1)*l2(i)
      r1(i,1) = r1(i,1) - b*l1(i)*l2(1)
80    continue

      do 81 i=1, n
      r1(1,1) = r1(1,1) + a1(n)*a2(i)*r2(n,i)
81    continue
      r1(1,1) = r1(1,1) - b*l1(1)*l2(1)
      return
```

```
end if
return
end
```

LINK.COM

```
$ DEFINE/USER CD MHD$LIBRARY:[V3E.CODE]
$ DEFINE/USER CDX MHD$LIBRARY:[V3E.CODEX]
$ DEFINE/USER ADP MHD$PROJECTS:[BPA.ADAPT]
$ DEFINE/USER DB MHD$LIBRARY:[V3E.DBOBJ]
$ DEFINE/USER TWOADT MHD$PROJECTS:[BPA.TWOTH.ADAPT.TEST2.TEST4]
$ DEFINE/USER MN MHD$LIBRARY:[V3E]
$ LINK/DEB/EXE=V3ETWOAD MN:MAIN, -
    DB:DGEN,LOLINE,DESVC,MODIFY,LSWITD,-
    CDX:SIMULA2X,-
    TWOADT:'p1',NETSOLX,UD2N,CONGPCN,MATSUB,MOLOAD,MN:ETMSPV3E/LIB
```

V3E.COM

```
$ set def mhd$PROJECTS:[BPA.twoth.ADAPT.TEST2.TEST4]
$ define/user MP mhd$PROJECTS
$! define/user sys$input sys$command
$! DEFINE/USER SYS$INPUT RNSTRM.V3B
$! DEASSIGN DBG$INIT
$ DEFINE/USER DBG$INIT DB.INT
$ run/debug MP:[BPA.twoth.ADAPT.TEST2.TEST4]v3Etwoad.exe
VRPRO
LOADFLOW
4
8
Q
89HSP_NDC.BSE
DYNDAT
89HSP_NDCO.DYN
LDYNDAT
RRESDA
Y
89HSP.OAP
N
SWITDAT
Y
?17MACHNO.FLT
89HSP.FLT
LSWITDA
Y
VRSWIT.SUM
INITIA
N
?;SLACK_GM20.0
SIMULA
QUIT
```

APPENDIX D

Sample Codes for the Simulations in Chapters 6 and 7

HOOK.FOR

```
C        EXAMPLE : SVC IS ADDED TO ...
C       This routine executes an interface to ETMSP - providing
C       "hooks" for to input signals into system and retrieve outputs.
C       The purpose is to allow simulation with nonconventional
C       or digital controllers. It also allows access to signals
C       during the run for the purposes of signal analysis, controls
C       or special outputs. Developed at Montana State University
C       in cooperation with the Bonneville Power Administration.
        SUBROUTINE HOOK

        INCLUDE 'PARX.INS/LIST'
        INCLUDE 'CNB8.INS/LIST'
        INCLUDE 'CNYVAR.INS/LIST'
        INCLUDE 'CNB4.INS/LIST'
        INCLUDE 'CNUDM.INS/LIST'
        INCLUDE 'CNVOLT.INS/LIST'
        INCLUDE 'CNMYVA.INS/LIST'
        INCLUDE 'CNNNEW.INS/LIST'
        INCLUDE 'CNSVS.INS/LIST'
        INCLUDE 'CNGEAR.INS/LIST'
        INCLUDE 'CNIPTR.INS/LIST'
        INCLUDE 'CNXSTA.INS/LIST'
        INCLUDE 'CNMSTA.INS/LIST'
        INCLUDE 'CNMDAT.INS/LIST'
        INCLUDE 'CNB5.INS/LIST'
        INCLUDE 'CNLFDA.INS/LIST'
        INCLUDE 'CNSTN.INS/LIST'
        INCLUDE 'CNNBLD.INS/LIST'
        COMMON /HKSTUFF/RNOIS
        REAL*4 CHEKC,DCON,TSPAST
        CHARACTER*12 BSNM(20),BSNM2
        INTEGER I,NT,BNUM(20), PT(20),POUT,PRNT,PNOW,II
        C -----------------------------------------------------------------
        C -------------------- START OF USER BLOCK 1 ----------------------
        C              - USER VARIABLE DEFINITIONS -
        REAL*4  REFORIG , VVOLTLO ,VVOLTL,VVOLTLRO,VVOLTLIO,ANGLELO
        REAL*4  VVOLTLON,VVOLTLN,VVOLTLRON,VVOLTLION,ANGLELON
        REAL*4 VVOLTLR,VVOLTLI,ANGLEL,DIFANGLE,VVOLTI2O,VVOLTR2O
        REAL*4 VVOLTLRN,VVOLTLIN,ANGLELN,VVOLTI2ON,VVOLTR2ON
        REAL*4 VVOLT2O,ANGLE2O,VVOLTR2,VVOLTIO,ANGLE2,ANGLEO,VVOLTI2N
        REAL*4 VVOLT2ON,ANGLE2ON,VVOLTR2N,VVOLTION,ANGLE2N,ANGLEON
        REAL*4 Y1,Y2,MW,MV,valo
        REAL*4 UNEXT,REPOWO,REPOW,SSPPEDO,SSPPED
        REAL*4  REPOWON,REPOWN,SSPPEDON,SSPPEDN,DIFFANG
        REAL*4  ADO(10,10),BDO(10),CDO(10),KO(10),KF(10),XHAT(10)
        REAL*4  UPR ,UCON,YOUT,YOBS,RNOIS,NOIS ,NOISO
        INTEGER  IREFPT , SVCNUM ,REFL,IMACC ,N,CCC,I1,I2
        INTEGER  IREFPT2,IREFPT3,IMACCN,IREFPT2N,IREFPT3N
```

```
      DATA I1,I2 /9587,7941/
      C --------------------- END OF USER BLOCK 1 -----------------------
      C -----------------------------------------------------------------
      C ******************** INITIALIALIZATION BLOCK ***********************
      C           OCCURS WHEN TS = H = 0.0
      IF ((TS .EQ. 0.0) .AND. (H .EQ. 0.0)) THEN
        OPEN(UNIT=21, FILE='HK.DAT',STATUS='NEW')
      C -----------------------------------------------------------------
      C --------------------- START OF USER BLOCK 2 ----------------------
      C           - ASSIGN NT, POUT AND EACH BSNM(I) UP TO I = NT. -
      OPEN(UNIT=22, FILE='HK2.DAT',STATUS='NEW')
      OPEN(UNIT=24, FILE='HK3.DAT',STATUS='NEW')
      OPEN(UNIT=23, FILE='THETA1.DAT',STATUS='NEW')
           POUT = 5
      NT = 4
      CGET NAMES FOR BUSES
      BSNM(1) = 'CANAD_MN 500'
      BSNM(2) = 'CALIF_TX 500'
      BSNM(3) = 'SOUTH_MN 500'
      BSNM(4) = 'COAST_MN 500'

      C --------------------- END OF USER BLOCK 2 ------------------------
      C -----------------------------------------------------------------
      PRNT = POUT
      TSPAST = -1.0
      CGET NUMBERS FOR BUSES
            DO 10 I=1,NT
         CALL CHKBUS(BSNM(I),BNUM(I))

      CCHECK FOR BUS # = 0, WHICH INDICATES A PROBLEM.
      CIF OK, THEN GET GENERATOR NUMBERS IN INTERNAL ORDER.
         IF( BNUM(I) .NE. 0) THEN
           PT(I) = NNEW(BNUM(I))
         ELSE
          WRITE(6,*) ' ERROR - PROBLEM WITH BUS NAME IN HOOK '
          CALL ASTOP('HOOK ERROR',16)
         ENDIF
      10  CONTINUE
C -----------------------------------------------------------------
C -------------------- START OF USER BLOCK 3 ----------------------
C  - GET THE INDICES OF ANY RELEVENT STATES USING PT(I) AS BUS INDEX -

C       THE NUMBER OF THE SVC IN DYNAMIC DATA FILE
        SVCNUM = 1
C       GET THE SVC Y-VARIABLE FIELD OFFSET
        IREFPT = JVSVR(SVCNUM)
C       GET THE SVC REFERENCE VALUE
c       REFORIG = BSVC(IREFPT + 2)
        REFORIG = RVREF(IREFPT + 1)
C       GET SVC BUS NUMBER(NEW)
C THIS GIVES  BUS NUMBER
      REFL = NROLD(SVCNUM)
```

```
C THIS GIVE MACHINE NUMBER
C       REFL = NTHG(PT(2))
C       GET SVC BUS INITIAL VOLTAGE
        VVOLTLO = ABS(VOLT(REFL))
C       GET STATIONN BUS VOLTAGE
C       IMACC = NTHG(PT(1))
C BUS NUMBER IS USED FOR THE KEELER
        IMACC = PT(1)
        IMACCN=PT(3)
C       IMACCN=NTHG(PT(3))
   IREFPT2=MYVAR(1,IMACC)
        IREFPT2N=MYVAR(1,IMACCN)
C       REPOWO=PDG(IREFPT2+2)
C       REPOWON=PDG(IREFPT2N+2)
        IREFPT3=MSTATE(1,IMACC)
        IREFPT3N=MSTATE(1,IMACCN)
C       SSPPEDO=XSTATE(IREFPT3+1)
C       SSPPEDON=XSTATE(IREFPT3N+1)
C       VVOLT2O = ABS(VOLT(IMACC))
        VVOLTR2O = REAL(VOLT(IMACC))
        VVOLTI2O = AIMAG(VOLT(IMACC))
C       VVOLT2ON = ABS(VOLT(IMACCN))
        VVOLTR2ON = REAL(VOLT(IMACCN))
        VVOLTI2ON = AIMAG(VOLT(IMACCN))
C       GET THE DIFFERENCE BETWEEN THE PHASE ANGLE OF THE BUSES
        ANGLE2O = (180.0/3.14)*ATAN2(VVOLTI2O,VVOLTR2O)
        ANGLE2ON = (180.0/3.14)*ATAN2(VVOLTI2ON,VVOLTR2ON)
C       VVOLTLRO = REAL(VOLT(REFL))
C       VVOLTLIO = AIMAG(VOLT(REFL))
C       ANGLELO =(180.0/3.14)*ATAN2(VVOLTLIO,VVOLTLRO)-ANGLE2O
        ANGLELO=ANGLE2ON-ANGLE2O
C       ANGLEO = ANGLELO
        valo=valin(51)
        OPEN (UNIT=10,NAME='SYSTEM2.DATA',TYPE='OLD')
        READ (10,*) N
        DO I=1,N
         READ (10,*) (ADO(I,J),J=1,N)
        END DO
        READ (10,*) (BDO(J),J=1,N)
        READ (10,*) (CDO(J),J=1,N)
        READ (10,*) (KF(J),J=1,N)
        READ (10,*) (KO(J),J=1,N)
        DO I=1,N
         XHAT(I)=0.0
c       KO(I)=ADO(I)
        END DO
        DO=1
        CCC=0
        UNEXT = 0
c call line power
C       call linepow(pt(5),pt(4),' 1',mw,mv)
```

```
C          - MAKE ANY INITIAL WRITES TO UNIT 21 -
      WRITE(21,1000)
C     WRITE(21,1001)
C     WRITE(21,1002)
1000  FORMAT('DATA WITH SVC ON MALIN 500 ')
1001  FORMAT(' ')
1002  FORMAT('3')
C --------------------- END OF USER BLOCK 3 ------------------------
C -----------------------------------------------------------------
C *************** END OF INITIALIZATION BLOCK ***********************
      ENDIF
C     CHECK FOR REPETE CALL AT SAME TIME BECAUSE OF DISCONTINUITY.
C     IF DCON = 1 THEN SET CONTROL ACTION FOR THIS CALL TO WHAT IT WAS
C     FOR PREVIOUS CALL OR JUST LET IT BE RECALCULATED.

      IF( TS .EQ. TSPAST) THEN
        DCON = 1.0
      ELSE
        DCON = 0.0
      ENDIF
      TSPAST = TS
C     END OF DISCONTINUITY CHECK
C     TEST FOR PRINTING
      IF (PRNT.EQ.POUT) THEN
        PNOW = 1
        PRNT = 1
      ELSEIF (PRNT.NE.POUT) THEN
        PNOW = 0
        PRNT = PRNT + 1
      ENDIF
      IF (DCON.EQ.1) THEN
        BACKSPACE(UNIT=21)
        PNOW = 1
        PRNT = PRNT - 1
      ENDIF
C -----------------------------------------------------------------
C --------------------- START OF USER BLOCK 4 -----------------------
C               - CONTROL ACTIONS -
C     CHANGE SVC REFERENCE VALUE
      IF ((TS.GE.0.0).AND.(TS.LT.0.2)) THEN
     UPR= .50
      ELSE
        upr=0.0
      ENDIF
C
C    NO FEEDBACK
c     BSVC(IREFPT+2) = REFORIG-UNEXT
C     IF ((TS.GE.4.0).AND.(TS.LE.5.0)) THEN
C     BSVC(IREFPT+2) = REFORIG + UNEXT + 0.0
C     ENDIF
      VVOLTL = VVOLTLO - ABS(VOLT(REFL))
C     GET ANGLE OF REF BUS
```

```
      VVOLTR2 = REAL(VOLT(IMACC))
      VVOLTI2 = AIMAG(VOLT(IMACC))
      ANGLE2 = (180.0/3.14)*ATAN2(VVOLTI2,VVOLTR2)
C     GET ANGLE OF BUS UNDER TEST
      VVOLTR2N = REAL(VOLT(IMACCN))
      VVOLTI2N = AIMAG(VOLT(IMACCN))
      ANGLE2N = (180.0/3.14)*ATAN2(VVOLTI2N,VVOLTR2N)
C     VVOLTLR = REAL(VOLT(REFL))
C     VVOLTLI = AIMAG(VOLT(REFL))
C     ANGLEL =(180.0/3.14)*ATAN2(VVOLTLI,VVOLTLR)-ANGLE2
      ANGLEL =ANGLE2N-ANGLE2
      DIFANGLE = ANGLEL-ANGLELO
C     DIFFANG=(DIFANGLE-ANGLEO)/0.01
C     ANGLEO = DIFANGLE
C     REPOW=PDG(IREFPT2+2)
C     SSPPED=XSTATE(IREFPT3+1)
C     REPOWN=PDG(IREFPT2N+2)
C     SSPPEDN=XSTATE(IREFPT3N+1)
c     Y1 = XSTATE(IREFPT3N+1)
      Y1 = DIFANGLE
      IF(Y1.GT.200) THEN
       Y1=Y1-360
      ENDIF
      IF(Y1.LT.-200) THEN
       Y1=Y1+360
      ENDIF
      Y2 =valout(4)
C     Y2= VVOLTL
      UNEXT=0
      IF (CCC.EQ.5)THEN
       DO I=1,N
       UNEXT=UNEXT+KF(I)*XHAT(I)
       END DO
       UNEXT=-UNEXT
       IF (TS.GT.0.0) THEN
        UCON=UPR+UNEXT
       ELSE
        UCON=UPR
       ENDIF
       YOUT=Y2
       RVREF(IREFPT+1) = REFORIG + UCON
       CALL OBSV(ADO,BDO,CDO,DO,N,KO,UCON,YOUT,XHAT,YOBS,TS)
       CCC=0
      ENDIF
      CCC=CCC+1
c call line power
      IF(RVREF(IREFPT+1).GT.100.0) THEN
       RVREF(IREFPT+1)=100.0
      ENDIF
      IF(RVREF(IREFPT+1).LT.-100.0) THEN
       RVREF(IREFPT+1)=-100.0
      ENDIF
```

```
C       call linepow(pt(5),pt(4),' 1',mw,mv)
C
        NOIS= 0.0
        DO II=1,12
          NOIS = NOIS + RAN(I1,I2)
        END DO
C
C Reset mean of distribution
        NOIS = NOIS - 6.0
        NOIS=NOIS
c       NOIS=0
        RNOIS=NOIS

C --------------------- END OF USER BLOCK 4 ----------------------
C ----------------------------------------------------------------
        IF (PNOW) THEN
C ----------------------------------------------------------------
C -------------------- START OF USER BLOCK 5 ---------------------
C                 - WRITE TO UNIT 21 -
        WRITE(21,2000)TS,RVREF(IREFPT+1),BSVC(IREFPT+2),YOBS,Y2,Y1
2000    FORMAT(1X,F7.4,2X,5(F10.5,2X))
        WRITE(22,3000) TS,ABS(VOLT(IMACCN)),VALIN(51)
     &      ,valout(4)
        WRITE(24,3000) TS,ABS(VOLT(PT(2)))
C       WRITE(23,4000) TS,(THETA1(I), I=1,10)
3000    FORMAT(1X,F7.4,1X,3(F10.5,1X))
4000    FORMAT(1X,F7.4,1X,10(F10.5,1X))
C --------------------- END OF USER BLOCK 5 ----------------------
C ----------------------------------------------------------------
        ENDIF
        RETURN
        END
```

OBSV.FOR

```
      SUBROUTINE OBSV(A,B,C,D,N,KO,U,Y,XHAT,YOBS,TS)
      IMPLICIT NONE
      REAL*4 A(10,10),B(10),C(10),KO(10),XHAT(10),XHATO(10)
      REAL*4D,U,Y,YOBS,XHAT2, ERR,TS
      INTEGERI,N,J
C     FIND OBSERVER OUTPUT (YOBS)
c        ko(1)=.265
c        ko(2)=-.7573
c        ko(3)=.7289
c        ko(4)=-.236
C        ko(5)=.658
C        ko(6)=-.1318
C        ko(7)=.0105
C        ko(8)=0
c        ko(1)=.0465
c        ko(2)=-.2251
c        ko(3)=.4409
c        ko(4)=-.4365
c        ko(5)=.2184
c        ko(6)=-.0442
C        ko(7)=KO(7)-30.6
C        ko(8)=KO(8)+6.88
         YOBS=0
         DO I=1,N
       YOBS= YOBS+C(I)*XHAT(I)
       END DO
C     FIND OBSERVER ERROR
      ERR = Y - YOBS
      DO I=1,N
       XHATO(I)=XHAT(I)
      END DO
C     GENERATE ESTIMATE OF X.

      DO I=1,N
       XHAT2=0
         DO J=1,N
        XHAT2 =XHAT2+ A(I,J)*XHATO(J)
         END DO
      XHAT(I)=XHAT2+B(I)*U+ KO(I)*ERR
      END DO


C        WRITE(*,600) T,Y,YOBS,ERR,U
 600  format(7(1X,1Pe9.2))
      RETURN
      END
```

OPEN.M

```
%
%------------------------------------------------------------------------------------------
%
% These codes are used to design a Linear Quadratic Gaussian Controller
% for a linear model of the power system.
%
%------------------------------------------------------------------------------------------
%
% load input and output data
n=6
load uytod
u=vref+.5;
%u(1)=u(1)-.5;
T=tt(2)-tt(1);
% load Prony identified model
load popdo
%rd=rd*26
% make transfer function
[num,den]=residue(r(1:n),e(1:n),0);
[a,b,c,d]=tf2ss(num,den);
% convert to discrete state space form
[ad,bd]=discr(a,b,T);
[numd,dend]=ss2tf(ad,bd,c,d,1);
%
% make system in observable canonical form
ado=[zeros(n,1) [eye((n-1),(n-1));zeros(1,n-1)]];
ado(:,1)=-[dend(2:n+1)]';
bdo=[numd(2:n+1)-dend(2:n+1)*numd(1)]';
cdo=[1 zeros(1,n-1)];
%
% find yhat of the identified model
yhat=[];
xk=zeros(size(bdo));
l=length(tt);
for k=1:l
 yk=cdo*xk;
 yhat=[yhat;real(yk)];
 xk=ado*xk+bdo*u(k);
end
%
% design lq on observable model
q1=cdo'*cdo;
r1=10
[kf,p]=dlqr1(ado,bdo,q1,r1);
%kf=acker(ado,bdo,p1);
ko=ado(:,1);
%koo=acker(ado',cdo',p2);
%ko=koo';
ko=dlqe1(ado,bdo,cdo,500,1);
xk=zeros(size(bdo));
```

```
xo=zeros(size(xk));
%xo(1)=.1;
ykk=[];yok=[];ukk=[];xkk=[];okk=[];
for k=1:l
 if tt(k)>=.1
  uk=-kf*xo;
 else
  uk=0;
 end
 % get the controlled output
 yk=cdo*xk +d*(uk);
 yo=cdo*xo +d*(uk);
 ykk=[ykk,yk];
 yok=[yok;yo];
 ukk=[ukk;uk];
 xkk=[xkk;xk'];
 okk=[okk;xo'];
 xk=ado*xk+bdo*(uk+u(k));
 xo=ado*xo+bdo*(uk+u(k))+ko*(yk-yo);
 %keyboard
end
```

CLOSE.M

```
%
%--------------------------------------------------------------------------------
%
% These codes are used to find a linear model for the power system
% using closed loop identified model.
%
%--------------------------------------------------------------------------------
%
% load Prony identified closed loop model
load popdc1
n=8
[numt,dent]=residue(r(3:n),e(3:n),0);
[at,bt,ct,dt]=tf2ss(numt,dent);
[at,bt]=discr(at,bt,.05);
ac=ado-ko*cdo-bdo*kf;
bc=ko;
cc=kf;
[numc,denc]=ss2tf(ac,bc,cc,0);
[ag,bg,cg,dg]=feedback2(at,bt,ct,dt,ac,bc,cc,0,1);
[numg,deng]=ss2tf(ag,bg,cg,dg);
%[ag,bg,cg,dg]=tf2ss(numg,deng);
[rd,ed]=residue(numg,deng);
ecr=log(abs(ed))/.05;
eci=angle(ed)/.05;
ec=ecr+i*eci;
l=length(ec);
for k=1:l
 %rc(k)=(rd(k)/ec(k))/(ed(k)-1);
end
rc=rd*20;
[numgc,dengc]=residue(rc,ec,0);
fr=linspace(0,1,200);
wv=2*pi*fr;
ss=i*wv;
for mm=1:200
 nume=polyval(numgc,ss(mm));
 dene=polyval(dengc,ss(mm));
 tg(mm)=nume/dene;
end
[aa,bb,cc,dd]=tf2ss(numgc,dengc);
%[ag,bg]=discr(aa,bb,.05);
%cg=cc;
yhatg=[];
xkg=zeros(size(bg));
l=length(tt);
for k=1:l
 ykg=cg*xkg;
 yhatg=[yhatg;real(ykg)];
 xkg=ag*xkg+bg*u(k);
end
```

OPENN.M

```
%
%-----------------------------------------------------------------------------------
%
% These codes are used to design a controller using inner/outer
% factorization method.
%
%-----------------------------------------------------------------------------------
%
% load input and output data file
n=6;
load uyto2
u=vref+.5;
u(1)=u(1)-.5;
T=tt(2)-tt(1);
% load Prony identified model
load pop2
%rd=rd*26
% make transfer fuction
[num,den]=residue(r(1:n),e(1:n),0);
[a,b,c,d]=tf2ss(num,den);
%gama=[bo bo*di;zeros(2,1) bi];
[numoo,denoo]=ss2tf(a,b,c,d,1);
% get discrete stste space form
[ad,bd]=discr(a,b,T);
[numd1,dend1]=ss2tf(ad,bd,c,d);
% separate inner outer transfer function
rnum=roots(numd1);
numi=conv([1 -rnum(2)],[1 -rnum(4)]);
%numi=conv([1 -rnum(1)],numi);
z2=1/rnum(2);
z4=1/rnum(4);
deni=conv([1 -z2],[1 -z4]);
%deni=conv([1 rnum(2)/1.2],[1 rnum(4)/1.2]);
%deni=conv([1 rnum(1)],deni);
numo=conv([1 -rnum(3)],[1 -rnum(5)]);
numo=conv(numo,[1 -rnum(1)]);
numo=conv(numo,deni);
deno=dend1;
[ai,bi,ci,di]=tf2ss(numi,deni);
[ao,bo,co,do]=tf2ss(numo,deno);
a=[ao bo*ci;zeros(2,6) ai];
b=[bo*di;bi];
c=[co 0 0];
c=-.0062*c;
gama=[bo bo*di;zeros(2,1) bi];
[numd,dend]=ss2tf(a,b,c,0,1);
%
%numd=real(numd);
%dend=real(dend);
% make system in observable canonical form
```

```
n=8;
numo=[0 numo];
ado=[zeros(n,1) [eye((n-1),(n-1));zeros(1,n-1)]];
ado(:,1)=-[dend(2:n+1)]';
bdo=[numd(2:n+1)-dend(2:n+1)*numd(1)]';
cdo=[1 zeros(1,n-1)];
% find yhat of the identified model
%[numm,denm]=ss2tf(ado,bdo,cdo,0,1);
%b=b/150;
yhat=[];
xk=zeros(size(bdo));
l=length(tt);
for k=1:l
 yk=c*xk;
 yhat=[yhat;real(yk)];
 xk=a*xk+b*u(k);
end
%
% design lq controller
q1=cdo'*cdo;
q1=1*c'*c;
r1=.5;
[kf,p]=dlqr1(a,b,q1,r1);
%kf=acker1(ado,bdo,p1);
%[kf,p]=lqr1(a,b,q1,r1);
%ko=ado(:,1);
%koo=acker1(ado',cdo',p2);
q=[1000 0;0 1];
[ko po]=dlqe1(a,gama,c,q ,1);
%ko=lqe1(a,b,c,1000,1);
%[adf,bdf]=discr(ao,bo,T);
[numdf,dendf]=ss2tf(ao,bo,co,do,1);
%
% make system in observable canonical form
n=6;
adof=[zeros(n,1) [eye((n-1),(n-1));zeros(1,n-1)]];
adof(:,1)=-[dendf(2:n+1)]';
bdof=[numdf(2:n+1)-dendf(2:n+1)*numdf(1)]';
cdof=[1 zeros(1,n-1)];
%ko=koo';
%q1=cdof'*cdof;
%r1=1
%kf=dlqr1(adof,bdof,q1,r1);
%ko=dlqe1(adof,bdof,cdof,2000,1);
%ko=[ko ;0;0];
xk=zeros(size(bdo));
xo=zeros(size(xk));
xo(1)=1;
ykk=[];yok=[];ukk=[];xkk=[];okk=[];
for k=1:l
 if tt(k)>=.1
  uk=-kf*xo;
 else
```

```
 uk=0;
 end
 % get controlled output
 yk=c*xk +d*(uk);
 yo=c*xo +d*(uk);
 ykk=[ykk,yk];
 yok=[yok;yo];
 ukk=[ukk;uk];
 xkk=[xkk;xk'];
 okk=[okk;xo'];
 xk=a*xk+b*(uk+u(k));
 xo=a*xo+b*(uk+u(k))+ko*(yk-yo);
 %keyboard
end
```