Experimental Modeling of a Hydraulic Load Sensing Pump using Neural Networks

A Thesis

Submitted to the College of Graduate Studies and Research

in Partial Fulfillment of the Requirements

for the Degree of Doctor of Philosophy

in

the Department of Mechanical Engineering

University of Saskatchewan

by

Xin Ping Xu

Spring, 1997

UNIVERSITY OF SASKATCHEWAN

College of Graduate Studies and Research

SUMMARY OF DISSERTATION

Submitted in partial fulfillment
of the requirements for the

DEGREE OF DOCTOR OF PHILOSOPHY

by

XU XIN PING

Department of Mechanical Engineering
University of Saskatchewan

Spring 1997

Examining Committee:

Dr. D.M. Gray	Department of Hydrology, Dean's Designate, College of Graduate Studies
Dr. R. Ford	Department of Agricultural Engineering
Dr. H. Wood	Department of Electrical
Dr. J. Wilson	Chair of Advisory Committee, Department of Mechanical Engineering
Dr. P. Ukrainetz	Department of Mechanical Engineering
Dr. G. Schoenau	Department of Mechanical Engineering
Dr. R. Burton	Co-supervisor, Department of Mechanical Engineering
Dr. C. Sargent	Co-supervisor, Department of Mechanical Engineering

External Examiner:

Dr. Nariman Sepehri
Department of Mechanical Engineering
University of Manitoba
Winnipeg, Manitoba

Experimental Modeling of a Hydraulic Load Sensing Pump using Neural Networks

The traditional method of modeling dynamics of a nonlinear hydraulic system is to develop mathematical describing equations based on "laws of nature". The development of these describing equations for a physical hydraulic system often requires that engineering intuition and a priori knowledge of the system be combined with mathematical properties of the equations. In addition, the problems of accurately measuring or defining physical parameters or coefficients have frequently restricted the interpretation of modeling results to specific operating points with very limited modeling accuracy. An alternative modeling approach is to establish or approximate mathematical relationships of a dynamic system based on observed input-output data. Neural networks have been a class of very attractive mathematical model structures that can be used to establish these mathematical relationships, because of their proven capabilities of approximating many nonlinear functions.

The overall objective of research in this study is to develop a neural network simulation package to assist designers in the simulation and configuration of hydraulic circuits. The specific objective of this thesis is to explore the capabilities of neural networks to approximate the nonlinear dynamics of a particular hydraulic component using experimental approach.

In this thesis, the use of partially recurrent neural networks with the conjugate gradient training algorithm to model a particular hydraulic load sensing pump was investigated. A simulation study was first conducted using "noise-free" data to examine the modeling errors in order to provide a clear insight into the mechanism of the modeling error accumulation over the transient state with a recurrent type of model structure. The established concepts and approach were then applied to experimentally modeling a load sensing pump. An experimental system was designed and constructed with particular attention paid to the design and generation of sufficiently rich input signals, and to the selection of an appropriate sampling rate. The data obtained on the testing of the load sensing pump dynamics are used in the training and testing of the neural models. The analysis and discussion showed that the training accuracy and the error accumulation were the two most critical factors in examining and interpreting the overall modeling accuracy.

It has been established through the work presented in thesis that a partially recurrent neural network is capable of approximating the dynamics of a hydraulic load sensing pump with very satisfactory accuracy.

The major contributions of this study are as follows: (1) the study identified the modeling error accumulation problem to be a major cause for the deterioration of the dynamic modeling accuracy, and suggested a means to effectively reduce the error growth in order to improve the modeling accuracy; (2) the applicability of the neural network approach to modeling a "real-world" hydraulic component using actual data was investigated, and practical constraints imposed by the actual hydraulic experimental testing facilities (not revealed by theoretical or simulation studies) were studied. The experimental implementation successfully established, from a practical point of view, the feasibility of the neural network approach to modeling a real hydraulic component, and suggested that the hydraulic data quality in terms of data precision and data distribution significantly affected final model accuracy.

BIOGRAPHICAL

Married, one child.

Education:

M. Sc., 1992, University of Saskatchewan, Saskatoon, Saskatchewan, Canada
B.S., 1982, Aero-Automatic Control Engineering, Northwestern Polytechnical University, PRC

Employment:

1997-date, Phoenix International, Fargo, ND
1982-1988, Xi'an Aircraft Company, Xi'an PRC

HONOURS

1992-95, University Graduate Scholarship, University of Saskatchewan
1992, Elmer Shaw Bursary
1997, appointed Net-Master, Fluid Power Systems and Technology Division, ASME
1996, funded by a grant from the Deere Company, Iowa

谨此献给我亲爱的母亲　徐廷湘

To my mother　Xu, Tin Xiang

Permission to Use

The author has agreed that the library, University of Saskatchewan, may make this freely available for inspection. Moreover, the author has agreed that the permission for extensive copying of this thesis for scholarly purposes may be granted by the professors who supervised the thesis work recorded herein or, in their absence, by the Head of the department or the Dean of the college in which the thesis work was done. It is understood that due recognition will be given to the author of this thesis and to the University of Saskatchewan in any use of material in this thesis. Copying or publication or any other use of the thesis for financial gain without approval by the University of Saskatchewan and author's written permission is prohibited.

Requests for permission to copy or to make other use of material in this thesis in whole or in part should be addressed to:

Head of the Department of Mechanical Engineering
University of Saskatchewan
57 Campus Drive
Saskatoon, Saskatchewan
Canada, S7N 5A9

Acknowledgments

The author wishes to express her sincere thanks to her supervisors, Dr. R. T. Burton and Dr. C. M. Sargent, for their support, encouragement, and enthusiastic participation during the course of this research and during the preparation of this thesis. The technical assistance of Mr. D. Bitner is also greatly appreciated.

The author gratefully acknowledges the sponsorship of Deere & Company, Iowa, U.S. for the experimental study in this project, and the financial assistance, in the forms of Doctoral Scholarship, provided by the College of Graduate Studies and Research, University of Saskatchewan.

The author would especially like to thank her parents, her husband, Pingbo, and her son, Xu Nan, for their understanding, encouragement and sacrifice during this study.

Abstract

The traditional method of modeling dynamics of a nonlinear hydraulic system is to develop mathematical describing equations based on "laws of nature". The development of these describing equations for a physical hydraulic system often requires that engineering intuition and a priori knowledge of the system be combined with mathematical properties of the equations. In addition, the problems of accurately measuring or defining physical parameters or coefficients have frequently restricted the interpretation of modeling results to specific operating points with very limited modeling accuracy. An alternative modeling approach is to establish or approximate mathematical relationships of a dynamic system based on observed input-output data. Neural networks have been a class of very attractive mathematical model structures that can be used to establish these mathematical relationships, because of their proven capabilities of approximating many nonlinear functions.

The overall objective of the research in this study is to develop a neural network simulation package to assist designers in the simulation and configuration of hydraulic circuits. The specific objective of this thesis is to explore the capabilities of neural networks to approximate the nonlinear dynamics of a particular hydraulic component using an experimental approach.

In this thesis, the use of partially recurrent neural networks with the conjugate gradient training algorithm to model a hydraulic load sensing pump was investigated. A simulation study was first conducted using "noise-free" data to examine the modeling errors in order to provide a clear insight into the mechanism of the modeling error accumulation over the transient state with a recurrent type of model structure. The established concepts and approach were then applied to experimentally model a load sensing pump. An experimental system was designed and constructed with particular attention paid to the design and generation of sufficiently rich input signals, and to the

selection of an appropriate sampling rate. The data obtained on the testing of the load sensing pump dynamics are used in the training and testing of the neural models. The analysis and discussion showed that the training accuracy and the error accumulation were the two most critical factors in examining and interpreting the overall modeling accuracy.

It has been established through the work presented in thesis that a partially recurrent neural network is capable of approximating the dynamics of a hydraulic load sensing pump with very satisfactory accuracy.

The major contributions of this study are as follows: (1) the study identified the modeling error accumulation problem to be a major cause for the deterioration of the dynamic modeling accuracy, and suggested a means to effectively reduce the error growth in order to improve the modeling accuracy; (2) the applicability of the neural network approach to modeling a "real-world" hydraulic component using actual data was investigated, and practical constraints imposed by the actual hydraulic experimental testing facilities (not revealed by theoretical or simulation studies) were studied. The experimental implementation successfully established, from a practical point of view, the feasibility of the neural network approach to modeling a real hydraulic component, and suggested that the hydraulic data quality in terms of data precision and data distribution significantly affected final model accuracy.

Table of Contents

List of Tables

List of Figures

Nomenclature

$y_n(j)$	jth neural network output
$y_{m,m}(k)$	model output in model-driven mode
$y_{m,p}(k)$	model output in plant-driven mode
$\Phi'[.]$	derivative of $\Phi[.]$
$\alpha^{(k)}$	step size at kth step, $k = 0, 1, 2, ...$
$\gamma^{(k)}$	the Polak-Ribiere formula scalar at kth step, $k = 0, 1, 2, ...$
$\Phi[.]$	nonlinear mapping function
δ_{ij}	1 if $i = j$; 0 if $i \neq j$
$\nabla J(.)$	gradient of $J(.)$
$\nabla J^{(k+1)}(.)$	gradient of $J(.)$ at k+1th step, $k = 0, 1, 2, ...$
ε	criterion function error bound
ω	load sensing pump shaft speed (r.p.m.)
δ	output error bound
τ	load sensing pump shaft torque (in lb, Nm)
θ	parameter vector
$\Delta\theta$	parameter θ deviation from their true values θ°
θ°	true value of parameter vector
$\Psi(k)$	nonlinear regressor
$(g^{(k)})^T$	transpose of $g^{(k)}$
$d^{(k)}$	direction vector at kth step, $k = 0, 1, 2, ...$
D_T	set of models
$e_f(k)$	fitting error or modeling error
$e_m(t)$	modeling error caused by experimental data noises
$e_s(t)$	modeling error caused by model structure differences
$e_{sm}(k)$	structure modeling error in the model-driven mode.
$e_{sp}(k)$	structure modeling error in the plant-driven mode

$f_m(.)$	general function
$g^{(k)}$	gradient vector at kth step, $k = 0, 1, 2, ...$
G(z)	error transfer function (see 5.3 in Chapter 5)
$J(.)$	object or criterion function
$J^{(k+1)}(.)$	object function at k+1 th step of iteration, $k = 0, 1, 2, ...$
L_i	number of neurons at ith layer
M	number of variables of object function or number of weights
$m(\theta)$	a model from the model set $M(\theta)$
$M(\theta)$	model corresponding to the parameter vector θ
N	number of training patterns or data pairs
$S(\theta^\circ, \Delta\theta)$	parameter space (an open ball) with center at the true parameter vector θ°, and radius $\Delta\theta > 0$
$n_m(w^\circ)$	a neural model from the model set $N_m(w)$ with w corresponding to fixed weights w°
$N_m(w)$	class of neural models
P	normalized pressure drop across an orifice
P_c	control pressure of the compensator in load sensing pump (psi, N/m^2)
P_l	load pressure (psi, N/m^2)
P_s	load sensing pump output pressure (psi, N/m^2)
Q	normalized flow rate through an orifice
Q_p	load sensing pump flow
$s_i(\phi)$	input to the nonlinear function at ϕth neuron of ith layer
$u(k)$	input at *kth* time step
$v_i(\phi)$	output of the nonlinear function at ϕth neuron of ith layer
$w^{(k)}$	weight vector at kth step, $k = 0, 1, 2, ...$
$w_{km}(\beta,i)$	weight connecting βth neuron at the kth layer to ith neuron at the mth layer
X	normalized opening of an orifice
$y_m(k)$	model output at *kth* time step
$y_p(k)$	plant or physical system output (desired output) at *kth* time step

z, z^{-1}	forward and backward shift operators

List of Abbreviations

NNARX	Neural Network based AutoRegressive eXogenous (inputs) ((see 4.3 in Chapter 4)
NNOE	Neural Network based Output Error (see 4.3 in Chapter 4)
NRMS	Normalized value of Root Mean Square
RMS	Root Mean Square
SISO	Single-Input-Single-Output system
WPR	Weighted Pseudo-Random signal (see 6.7.2 in Chapter 6)

Chapter 1
Introduction

1.1 Problems of Traditional Nonlinear Modeling Approach

In recent years, the modeling and simulation of actual hydraulic components or systems have become an increasingly important method used to analyze and optimize the system performance in the hydraulic system[1] design. Traditional methods used to model the nonlinear behavior of a hydraulic dynamic system (indeed all systems) often involve mathematical modeling procedures based on the observation of physical relationships among the associated components. Recent advances in computing techniques and mathematical modeling of nonlinear dynamic systems have resulted in powerful computer simulation packages, [Richards, C. R. et. al., 1989; Hopsan, 1985], MATLAB, SIMULINK and EASY 5, just to name a few, which are playing an increasingly important role in the simulation of complicated nonlinear hydraulic systems.

However, the derivation of a suitable model for a practical component requires that engineering intuition and a priori knowledge of the system behavior be combined with mathematical properties of the model. In addition, all parameters, such as those describing physical properties of components and of the fluid medium, or coefficients used in empirical formulas must be supplied or obtained experimentally. This mathematical modeling process has often turned out to be a formidable task, because of not only the resulting complicated mathematical expressions of nonlinear model structures that often require much computational effort to solve for a practical solution,

[1] Unless otherwise specified, the term "system" used throughout this thesis is referred to as an actual physical system.

but also the physical constraints that make experimental determination of some critical parameters or coefficients difficult and imprecise, especially when a direct access to the parameters of interest becomes limited or impossible. One example is the determination of the Coulomb friction coefficient in the friction forces which exist in a hydraulic actuator. Any degradation of accuracy in the measurement or determination of the parameters would affect the modeling accuracy and hence reduce the developed confidence in the model quality. Linearization techniques simplify nonlinear modeling procedures considerably, but, at the same time, impose a validity constraint as the linear model obtained is only valid over a very limited region of the operation.

1.2 Alternative Approach - System Identification

An alternative to mathematical modeling is system identification. System identification is an experimental approach to the determination of the model of a real system based on the observed input-output relationships. In parametric system identification[2], models take forms of mathematical expressions, such as differential or difference equations, with variable parameters. Model parameters that may or may not reflect any physical considerations in a real system are adjusted to fit the model into input-output data measured from a properly designed identification experiment. A parametric identification procedure in general involves four basic stages: data acquisition, model structure selection, parameter estimation and model validation.

A model structure can be loosely defined as a certain form of mathematical expression with parameters being variables. As the parameters span a set of feasible values, the model represents a set of models, or a class of models, or a model structure (these three terms can be interchangeably used). When all the parameters are given specific values, the model set is reduced to a model. It is no doubt that the most important and, at the same time, the most difficult choice is the selection of a suitable set

[2] The interpretation of the "parameter" system identification is based on [Soderstrom, T. and Stoica, P., 1989]

of models within which a model that is 'equivalent' to the system under identification exists. Once a model structure is properly chosen, the task of parametric identification then becomes a problem of determining or estimating the values of the model parameters through applying any of the existing algorithms [Soderstrom, T. and Stoica, P., 1989; Ljung, L., 1987].

Linear dynamic system identification has been reasonably well understood, and parameter estimation methodology has been systematically established [Astrom, K. J. and Eykhoff, P., 1971], because simple linear forms of model structures have been assumed. In the field of nonlinear dynamic system identification, the choice of a suitable model structure, at present, still remains as a challenging topic in this area. Indeed, it is, in general, not practical to talk about the identification unless a specific model structure is imposed. The difficulty in selecting a model structure lies in the fact that there is still a lack of unified mathematical theory that can be directly applied for representing (or modeling) various dynamic characteristics of nonlinear systems.

Several approaches to the determination of model structures for nonlinear dynamic systems have been studied in the past [Haber, R. and Unbehauen, H., 1989]. A popular approach is based on the decomposition of the whole model into several simpler subsystems. Depending on the forms of each subsystems, and on how the subsystems are arranged (addition, multiplication, in parallel, in series etc.), several well-known classes of models are formed through this approach, such as the block-oriented models consisting of static nonlinear subsystems and dynamic linear subsystems, and hierarchical multilevel models (an example of this model structure is GMDH—Group Method of Data Handling, also called self-organization method [Farlow, J. S., 1984]). Another well-known approach that can model a wide class of nonlinear dynamic systems is 'linear-in parameters'. That is, the parameters to be estimated appear linearly in a nonlinear difference equation, while input and output variables appear nonlinearly. Several existing nonlinear model structures can be categorized into the model of this class, such as Volterra series expansion with finite order, the Hammerstein model that is composed of a

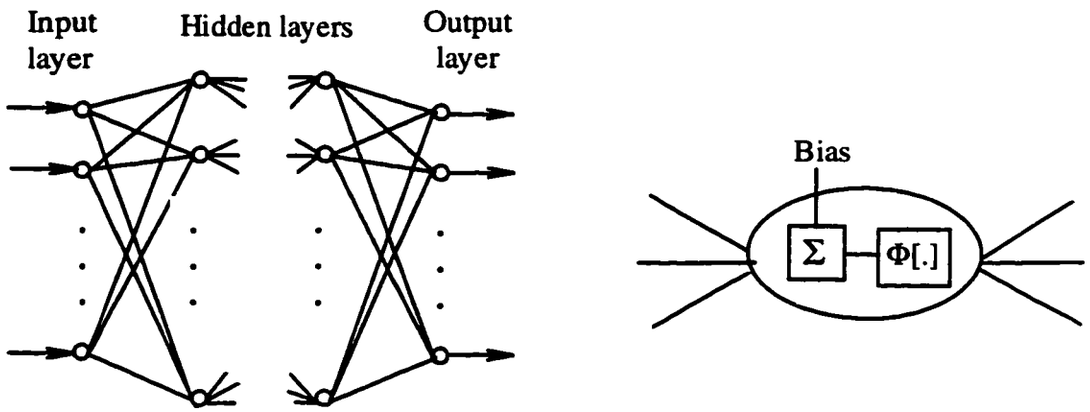
static nonlinear gain followed by a linear dynamic system [Hsia,T.C., 1977], and function link network (FLN) which lumps various nonlinear functions into the network input vector and the network connection is in a linear fashion [Chen, S. and Billings, S. A., 1992]. In all above approaches, however, there is still a need for the determination of specific forms of subsystems and their connections through preliminary test or data analysis, or the determination of nonlinear function terms in which input and output variables appear in the 'linear-in parameters' approach.

1.3 Neural Network Approach and Its Current Applications

Artificial neural networks, as a class of very attractive mathematical model structures, have received increasing attention in the nonlinear system identification area. An artificial neural network is a massive net that consists of a number of identical computing units referred to as neurons or nodes. The morphology of a neural network can vary depending on the way the neurons are interconnected and the operations performed at each neuron. In a multi-layered feedforward network, neurons are arranged in layers, as shown in Figure 1.1. All neurons in a layer are fully connected to the neurons in the adjacent layers, and there are no connections between the neurons in the same layer. Data information is passed through the network in such manner that the outputs of the first-layer neurons become the inputs to each neuron in the second layer and so on. A connection between two neurons is referred to as a weight, a variable parameter that mathematically represents the strength of the connection. The operations at each neuron are usually a summation of weighted inputs to that neuron and then a nonlinear transformation (mapping), $\Phi[.]$. In this study, a sigmoidal function has been adopted unless stated otherwise. Each neuron may have a bias associated with it.

Multi-layered feedforward neural networks have been proven theoretically to be able to approximate nonlinear functions to any degree of accuracy as long as there are enough neurons at hidden layer(s) (the layer(s) between input and output layers) [Hornik, K., Stinchcombe, M., and White, H., 1989]. Such a neural network certainly affords an

alternative to model structures for complex nonlinear system identification. The neural network should be viewed as a special class of models which are functional representations of nonlinear systems. It is the activation function (continuous, bounded and nonconstant) at each node and the multilayer feedforward architecture that ultimately give the network the nonlinear approximation ability [Hornik, K., 1991]. The appeal of the neural net model is its flexibility, one of the desirable properties in model quality. That is, it can be universally applicable, with its unique structure, for approximating many different and complex types of nonlinear functions, provided there is a sufficient number of neurons at hidden layer(s). Using a multilayer feedforward neural net as a center element, various types of dynamic neural models can be formed to capture the dynamic features of the unknown system.



(a) A multi-layered feedforward neural network (b) Structure of a neuron

Figure 1.1 Architecture of a multi-layered feedforward neural network.

The use of neural networks for static nonlinear system identification has been recently studied extensively in many areas for various purposes. Like any other mathematical model structures, an obtained neural net model for a specific plant must be justified through validity tests. The criteria for the validity tests should be formulated to reflect the intended purpose of the model, and the required accuracy which usually is application dependent (subjective). Current applications of neural network models in the

area of nonlinear dynamic systems can be categorized into three major types by the purpose of using the models: prediction, simulation and trajectory tracking.

The author has noticed from the literature review that the terms, prediction, simulation and tracking, under the subject of identification, have been frequently used, with little clarification, in the extensive literature for the diverse applications of system identification and control. Although often the application itself may imply the physical considerations that these terms are referred to, clarification on the terminology is necessary to avoid the ambiguity in the description and formalization of the identification problems in this thesis. With reference to Figure 1.2, a brief interpretation of the term simulation which is the main subject of this thesis will be given at first. The interpretations of prediction and tracking will also be included for classification and completeness.

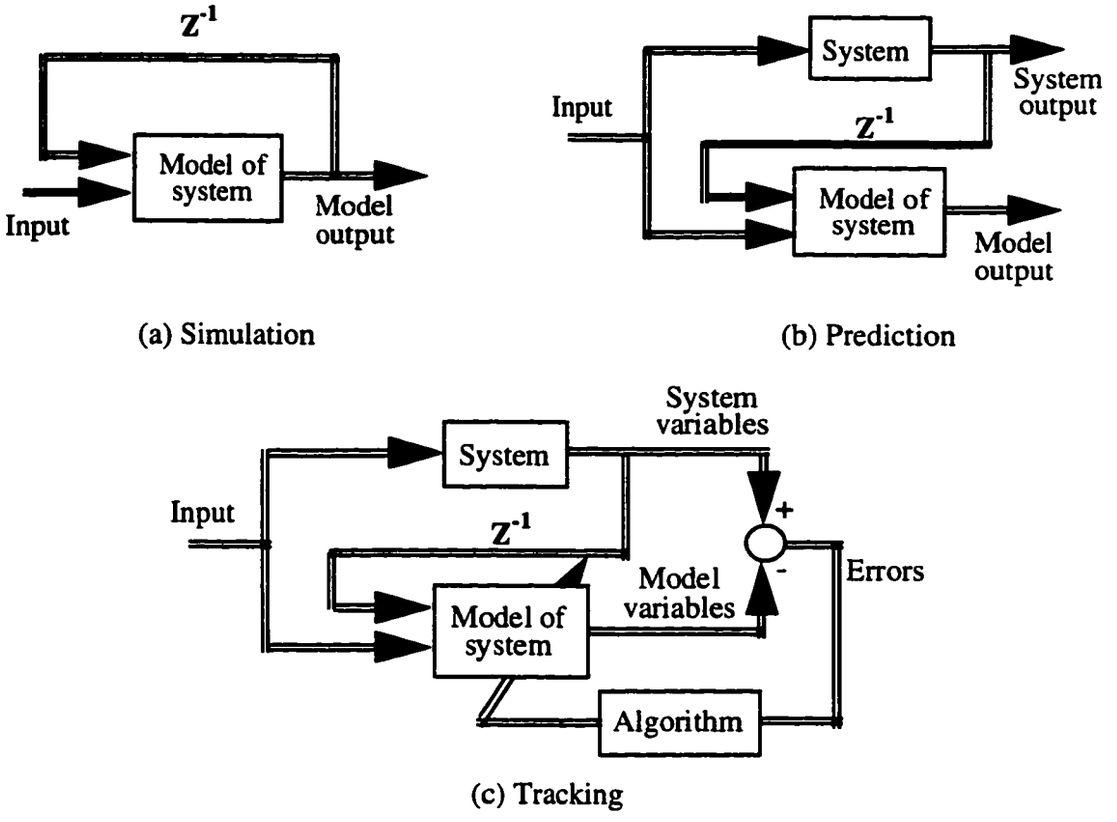


Figure 1.2 Uses of developed models for different purposes

Simulation:

The simulation model will be used independent of the real system after the model is established. That is, the current output of the model is evaluated from the current input and past inputs, and **past model outputs**.

$$y_m(k) = f_m(u(k-i), y_m(k-l)) \qquad (1\text{-}1)$$

where $f_m(.)$ is a general function representation with finite dimensions. (k-i) and (k-l) are time delays, and i=0, 1, 2, ..., and l=1, 2, ... The models essentially work as a simulator, or substitute of the system, and can be used both 'on-line' for prediction in control and 'off-line' for simulation studies.

Prediction:

The estimated model is used to predict the system output based on the past inputs and **past system outputs** that are available up to the time instant:

$$y_m(k) = f_m(u(k-i), y_p(k-l)) \qquad (1\text{-}2)$$

where $i \geq 1, 2, ..., l \geq 1, 2, ...$. The models can only be used 'on-line'.

Note that a simulation model can generally work as a prediction model, but not vice versa.

Tracking:

Tracking essentially is to modify the parameters of a selected model simultaneously while estimating the system's variables of interest. Parameters of the model are adapted instantaneously driving the model to follow the desired trajectory of the variables. Tracking (or sometimes called parameter estimate in control) is different from prediction and simulation in that parameters of the model are always changing when the model is in use.

Most successful applications using neural net models in the area of dynamic nonlinear systems have been limited mainly to applications for the purposes of prediction

and trajectory tracking in controls [Levin, A. U., 1993; Dracopoulos, D. C. and Jones, A. J., 1993]. A model that gives satisfactory results under the prediction mode may not necessarily work equally well in the simulation mode for the same system. In general, modeling of a dynamic nonlinear system for simulation purpose imposes the most restrictive constraints on the accuracy of the model, because the modeling errors (differences between the system output and its model output) associated with past model outputs are backpropagated through the feedback paths (the terms of $y_m(k-l)$, $l=1,2, \ldots$ in Equation (1-2)) and could then be cumulated, resulting in excessive overall fitting errors. If measurement noise is a major concern, then the accuracy problem is compounded. Work in the use of neural net models in the nonlinear dynamic system identification area for simulation involving high modeling accuracy has been rarely reported so far in the published literature.

1.4 Neural Net Simulator

An established neural net model for the simulation of a nonlinear dynamic system is required to be able to produce, ideally, identical outputs to those from the system when subjected to different input signals other than those used for the estimation of the model parameters. A complete identification process of a neural network consists of two stages: training and testing. Training adjusts all parameters or weights of the net according to a learning algorithm until an acceptable correspondence between the input and output of the neural net is achieved. Testing is to assess the neural net generalization property (interpolation capabilities) and the modeling accuracy, in which all parameters of a neural network are fixed and new input-output data pairs are fed into the trained neural net. Only when the test results meet the requirements on the accuracy and generalization (the two aspects actually are closely related to each other), can the obtained model be accepted as a simulator of the real component for which the network has been trained. Such a neural simulator can then be used as a substitute for a real nonlinear component in simulation studies. Furthermore, a neural simulator for the simulation of a complete circuit can be built by an integration of a number of neural

simulators for the corresponding physical components that are intended to be included into the circuit.

1.5 Motivation and Objective of This Thesis

In an effort to circumvent the problems in the traditional modeling approach, modeling a particular hydraulic nonlinear system using the identification approach with a specific neural net model structure is to be investigated. The overall objective for this particular research is to develop a neural network simulation package to assist designers in simulation and configuration of hydraulic circuits. The development of such a simulation package is based on the consideration that a hydraulic circuit configuration is now more likely being performed by component selection rather than component design, since various types of hydraulic components are now available in stock or can be found in manufacturer's catalogues.

As has been found from a literature review, research in using a neural model as a simulator of a nonlinear dynamic system still remains an open and challenging area. In this regard, the first and necessary step toward the overall objective would be to study the feasibility of a neural net simulator. The specific objective of this thesis is, therefore, to explore the capabilities of a neural network to approximate dynamic characteristics of a hydraulic system by developing a dynamic neural model for a particular hydraulic component using experimentally measured input-output data.[3]

The component chosen for this project is a load sensing hydraulic pump - a variable displacement (flow) pump which is designed to control flow independent of loading conditions in an energy efficient manner. Because it is a complex control system, it has been a difficult component to model using traditional approaches. It is believed that if the feasibility of using a neural network simulator can be established on this particular

[3] A hydraulic component was chosen only because of the author's familiarity with the subject area. Other devices in mechanical and electrical areas could also have been chosen.

component, then other complex components and systems (not necessarily in the hydraulic area) may also be good candidates for neural network modeling. It is the intention of this particular research study to provide the basic ground work upon which applications can be expanded upon.

1.6 Organization of This Thesis

Chapter 2 is devoted to the formalization of the identification problem using neural networks through the approximation approach. It is emphasized from a practical point of view that a model is only an incomplete and simplified representation of a real system for a particular purpose, and shall be pursued as a best approximate of the system in an engineering sense.

Chapter 3 presents the neural network training algorithm based on the error backpropagation and conjugate gradient method. The numerical properties of fast convergence and stable solutions of this algorithm are demonstrated through approximating the nonlinearities of a theoretical and a physical model.

In Chapter 4, partially recurrent neural networks and two nonlinear regression schemes are introduced for modeling dynamics of several nonlinear systems. A simulation study is performed to model theoretical nonlinear dynamic systems using noise-free data for the purpose of examining the modeling structure error behavior.

Chapter 5 further analyzes the observed structure error. The analysis reveals the mechanism of modeling error growth that is associated with the recurrent model structure, and suggests a means to improve the modeling accuracy.

In Chapters 6 and 7, the application of established concepts and approach to the modeling of the load sensing pump using experimental data is presented. Chapter 6 describes the operating principle of the load sensing pump and the justification of its

model configurations, and the identification experimental system including the software programming and hardware installation of a computer controlled data acquisition system. Some practical constraints in the experimental implementation are addressed. In Chapter 7, the modeling results for the two different pump model configurations are presented. The modeling accuracy and generalization property of the neural networks are examined through the model validity testing, and the effects of data noise and error accumulation on the modeling accuracy are discussed.

Chapter 8 gives conclusions and recommendations for the future work.

Chapter 2

Statement of Practical Identification Problems

In this thesis, a stated objective was to model an actual hydraulic system using a neural network. The subject of this study is the practical identification of a hydraulic component. The problem of "identification" has been extensively researched over the past decades; indeed, the framework in establishing the validity of the "classic identification approach" has been formally defined (Soderstrom, T. and Stoica, P., 1989; Ljung, L., 1987; Bellman, R. and Astrom, K. J., 1970). In this study, the approach used to identify a physical component is based on neural network technology and approximation concepts. The foundation of neural network technology has been equally well researched over the decades (Zurada, J. M., 1992). However, a formal relationship between the neural network modeling approach and the classic identification approach has been marginally addressed in the literature (Billings, S. A., Jamaluddin, H. B. and Chen, S.,1992).

This chapter is intended to formalize the practical identification problem in relation to the classic identification approach. This formalization is based on an approximation concept, and is necessary to provide theoretical fundamentals and a systematic framework within which the practical identification problem is approached. The practical identification approach established has naturally led to the validity method for the justification of neural network model quality which is different from that used in classical identification approach.

The relevant definitions and approaches which have appeared in a majority of identification literature are first reviewed and examined. This review shall provide an

overview on the classic approaches to the identification problems, which are theoretically based on the concept of "identifiability". In addition, this chapter shall provide a comparison of the classic approach to the approximation approach from a practical point of view. The formalization of practical identification problems using an approximation concept emphasizes that, in reality, a model of a real system should be pursued as a best approximation of the system in an engineering sense.

2.1 Introduction

Although the term identification has different connotations or emphases in different disciplines, an identification problem using parametric models can be generally stated as follows:

> Given a physical system under test, select (or given) a group of inputs and a class of models.
> Find a member from the model class based on the input-output observations. The member is equivalent to the system in the sense that the model responses to all inputs are, ideally, identical to those of the system.

The original concerns with the existence of solutions to such a model set led to concept of "identifiability". Definitions and applications of identifiability in many areas have been found in a considerable amount of publications [Nguyen, V. V. and Wood, E. F., 1982; Gustavsson, I., Ljung, L. and Soderstrom, T., 1977]. In most of the literature, identifiability is approached by assuming a mathematical equation (a model set) representing exactly the true structure of the system, and then searching for the conditions under which the equation parameters can be uniquely determined. The key to this approach is that the *equivalence* of the estimated model to the system is established through the convergence of the parameter estimate to the *true* system parameters. This approach is referred to as a "theoretical identification approach" since it uses the hypothesis of the existence of real system equations (real structure and real parameters).

In practice, on the other hand, an actual system often is nonlinear, complex, and can never be completely known. A model is just a simplified representation or approximation of a real system, and its outputs cannot be identical to that of the real system. The solution of model parameters can therefore never be unique. The identification problem, in the practical sense, becomes one of finding a member from the given model set. This member approximates reasonably well the system, in the sense that the differences between the model and the system responses to all inputs fall and stay within a given tolerance. Such a practical identification approach (herewithin defined as the term "approximation") is not in contradiction with theoretical identifiability concepts, but rather a view of identification problems from a real-world perspective. The approximation approach is more concerned with modeling accuracy, rather than the uniqueness of solutions to the model parameters.

In the following sections, the theoretical approach (identifiability) will be first briefly reviewed, and then the approximation concept used in practical identifications will be described and formalized. Finally, identification problems using neural networks will be stated through the approximation approach.

2.2 A Review on the Theoretical Identification Approach — "Identifiability"

This section is included here as it is necessary to understand the framework used to formalize practical identification problems through the approximation approach.

The concept of parameter identifiability with perfect data observations was originally introduced and referred to as structure identifiability by Bellman and Astrom [Bellman, R. and Astrom, K. J., 1970] for compartmental models in biomedical applications where the model parameters were important. The basic assumption used in structure identifiability problems was that the given model set represented the true structure of the real system under study, and the task of identification would be therefore

to determine the true values of the parameters from the data observations. The central problem in the identifiability analysis is the uniqueness of the solutions to the model parameters. Godfrey [Godfrey, K., 1983] has abstracted the concept of the identifiability from Bellman and Astrom's, and others' work, and termed it as deterministic identifiability. This is quoted as follows:

> "(I) The parameters of an assumed model can be estimated uniquely and the model *is globally (uniquely) identifiable* from the experiment.
> (ii) Any of a finite number of alternative estimates for the model parameters fits the data and the model is *locally identifiable*.
> (iii) Any of an infinite number of estimates fits the data and the model is *unidentifiable* from the experiment."

Other approaches to structure identifiability include the one presented by Grewal and Glover in their paper [Grewal, M. S. and Glover, K., 1976].

There has been some effort made over past years to broaden the definition of identifiability so as to introduce the considerations of some practical aspects (data noise). Godfrey [Godfrey, K., 1983] defined the identifiability in which imperfect data were considered as "numerical" identifiability in contrast to the "deterministic" identifiability that assumed perfect data observations. Hadaegh and Bekey [Hadaegh, F. Y. and Bekey, G. A., 1985] considered structure error and proposed the concepts of "near-equivalence". The near-equivalence problem is concerned with determining if, for any allowable parameter deviation $\Delta\theta$ from their true values θ°, and an allowable output error bound δ, the parameters θ stay in the neighborhood of their true values $\theta \in S(\theta^\circ, \Delta\theta)$, when the model and system responses stay within the given bound, $\| y_p - y_m \| \leq \delta$. $S(\theta^\circ, \Delta\theta)$ is an open ball with center at the true parameter vector θ°, and radius $\Delta\theta > 0$. The definition of near-identifiablity based on the model-system near-equivalence relationship was then established. A class of model is said to be locally (or globally) nearly identifiable at θ°, if the model $m(\theta^\circ)$ is nearly equivalent to the system and the criterion

function has an isolated local (or a unique) minimum at $\theta = \theta^{\circ}$. Although Hadaegh and Bekey still used an assumption of existence of true parameters and considered the model parameters in the neighborhood of these *true* parameters, they have introduced the approximation concept in their work.

Soderstrom and Stoica have separated the identifiability concept into *system identifiability* and *parameter identifiability* [Soderstom, T. and Stoica, P., 1989]. In their work, standard white noise was introduced into the data and error model structures were considered. A concept of a "true system" was introduced to denote a mathematical description of the real system. A true system was used as a "fiction of a real-world system" for the purposes of theoretical analysis, for the comparisons among the different identification methods under various circumstances, and for the generation of the observed data in simulation studies. The identifiability analysis was then performed to determine whether and how an "unbiased" or "consistent" estimate can be obtained for a true system based on the data observations. An estimate is biased if its expected value deviates from the true value, and an estimate is consistent if the estimate tends to the true parameter as the number of data point tends to infinity. The concept of identifiability is hence approached by relating the identifiability property to consistency of the parameter estimates in some stochastic sense.

Given a model structure, a true system and the experimental conditions, the system is said to be *system identifiable* if parameter estimates, θ, approach to the true parameter set D_T as the number of data points N approaches to infinity. This is expressed as:

$$\theta \rightarrow D_T \qquad \text{as } N \rightarrow \infty \qquad \text{(with probability one)} \tag{2-1}$$

D_T is a parameter vector set that consists of those parameter vectors for which the model structure gives a perfect description of the true system. If D_T consists of more than one point, then for system identifiability, it must be ascertained that the shortest distance between the estimate and the set D_T tends to zero as N tends to infinity, expressed as:

$$\lim_{N\to\infty} \inf_{\theta\in D_T} \|\theta - \theta^o\| = 0 \qquad (2\text{-}2)$$

The system is said to be *parameter identifiable* if it is system identifiable and D_T consists of *exactly* one point. For a parameter identifiable system, the estimate will be unique for large amount of data and also consistent.

In this approach, the quality of identification is assessed by the convergence of the parameter estimates to the *true* parameters of the system. That is, the validity of an estimated model is justified by verifying that the parameter estimates are unbiased or consistent using various techniques. A consistent estimate requires that the system is parameter identifiable, which, in turn, indicates that the model structure perfectly represents the system to be identified. This validation method is relevant only when the true system can be expressed with some well-defined mathematical rules from which the data for the estimation are generated.

In the real-world identification process, however, the system to be identified can never be completely known and perfectly described by a set of mathematical equations with finite dimensional parameters. A mathematical model is partial and limited description of the reality, and a representation of some aspects of a real system. A model is not equal to a system, no matter how appropriately the model structure is selected. There always exist differences in the responses of a system and its estimated model, because of the differences in the structure. In this context, model parameter estimates will be biased with respect to the "true parameters" of the real system. A model is only an approximation of a real system, and an identification problem is essentially an "approximate problem" when viewed from practical perspective.

2.3 Practical Identification Approach — Approximations

When a practical identification problem is approached from approximation considerations, it becomes a problem of finding a mathematical model based on the experimental data such that the estimated model is equivalent to the real system under

test in the sense that the model gives a "good enough" approximation of the system in its response to all inputs. Several questions arise: "How good is "good enough" ?". If a criterion of goodness is specified, then "Is the model set selected able to meet this criterion ?". The answer to the first question is essentially application dependent. It is the users' specification that will provide a measure of the goodness of the approximation. The second question concerns another aspect of the approximation problem: an achievable, practical modeling accuracy which reflects the inherent approximation capability of a given model structure for a given real system. The approximation problem is thus approached from either of two ways: examine whether the modeling accuracy has met the user requirements, or study the inherent approximation ability the model possesses for the given system. Approximation analysis, therefore, concerns the quality of the approximation measured by the extent of the agreement between the real system outputs and model-generated outputs.

In the formulation of an identification problem using an approximation approach, the hypothesis of the existence of a true system or true parameters is not used. Assume a properly selected dynamic model structure $f_m(\theta)$ that approximates a unknown system y_p at θ^*:

$$y_p(k) = f_m(\theta^*, u(k), u(k-1), u(k-2), ..., u(k-m), y_p(k-1), y_p(k-2), ..., y_p(k-n)) + e_f(k) \quad (2\text{-}3)$$

such that a properly formulated criterion function that measures the norm distance between the responses of the model and system takes a minimum at θ^*. θ^* is therefore the best parameter vector with respect to the chosen model structure, i.e. among all the parameter sets of the chosen model structure, θ^* makes the model $f_m(\theta^*)$ the best approximation of the real system. The term $e_f(k)$ in Equation (2-3) is a general representation of the residual between the system and the model.

The model structures that satisfy Equation (2-3) will not be unique. which implies the nonuniqueness of the solutions to an approximation problem. Parameter

estimates must also be biased since all of the best parameter set for the corresponding model sets are not equal to the true parameters of the real system.

In determining a model from the model set, a practical identification procedure generally involves the searching for a set of parameters that minimizes the criterion function while checking whether the model and system responses have fallen and stayed within an allowable bound. In general, a practical identification problem (or an approximation problem) can be stated as follows:

> Given an unknown real system, a criterion function J(.), and error bound ε associated with the criterion function, and δ associated with outputs, then select a model structure $M(\theta)$ and a class of inputs.
>
> Decide a model $m(\theta)$ from the model set $M(\theta)$ such that
>
> $$J(\theta) \leq \varepsilon \tag{2-4}$$
>
> and
>
> $$\| y_p(k)-y_m(k) \| \leq \delta \tag{2-5}$$

then the model is an approximator of the system, or the model and system are considered to be equivalent in the sense that their responses to all inputs stay within a tolerable error bound.

If the parameters of a model represent physical meanings, the problem of whether the best values of these unknown parameters can be determined from observations of the system, or more specifically, whether the model structure does not, by itself, prevent these parameters from being identified would become a major concern. Other factors, such as experimental conditions and estimate algorithms will also affect the quality of parameter identification, but in general, parameter identification problems are concerned under the assumption of the best circumstances with regard to the data quality and estimate algorithms. There may arise some cases where different parameter estimates give identical model responses to the same inputs. The key question is whether those solutions can be distinguished from each other. If several solutions are

sufficiently far apart, then one can isolate a valid solution from extraneous solutions based on the principle that a valid solution must be restricted within a realistic and meaningful domain. Thus the parameters with the chosen model structure are considered locally determinable (identifiable), and the model estimated is acceptable.

The concept of parameter identifications in approximations can be approached in a way similar to that in theoretical identification problems, except that the concept of the "true parameters" of the real systems is no longer relevant, and the "best parameters" for the chosen model structure will be used instead. Without mathematical proof, the problem of determination of the best unknown parameters can be stated as follows:

A model structure is said to be locally (or globally) parameter identifiable at θ^*, if, in addition to satisfying Equations (2-4) and (2-5), the criterion function $J(\theta)$ has an isolated local (or a unique) minimum at $\theta = \theta^*$.

There may be more than one model structure that all give equally good approximation results for the same real system (i.e. they all meet the criterion of the required modeling accuracy). The selection of one model among those competitive modes will be influenced by many other factors. Two of the most important factors are simplicity of the model structure (the model formulation is simpler, and the number of parameters is less), and less effort to compute the model, which is associated with the algorithm complexity and the criterion function formulation.

2.4 Statement of Identification Problems Using a Neural Network Approximator

Neural networks when used to "simulate" physical systems represent another form of the identification approach. Neural networks can adopt many morphologies with regard to the types of nonlinear function at each neuron, the ways of connections, the number of neurons and the number of layers. By the concept of model set, neural network-type of model structures can be further divided into many different sub-classes.

and each class of neural models corresponds to a specific type of morphology. Let $N_m(w)$ denote one sub-class of models, where w stands for the neural net weight vectors (or model parameter vectors). When w takes fixed values w°, $N_m(w)$ becomes a neural model of the chosen model set, $n_m(w^\circ) \in N_m(w)$.

Based on the approximation approach described above, the identification problem using a class of neural network models to approximate a real system can be formulized straightforwardly as follows:

Given:

an unknown system,

a class of input-output data observations,

a class of model $N_m(w)$,

a criterion function J(w),

an error criterion ε,

an output error bound δ,

find a model $n_m(w^\circ) \in N_m(w)$ such that

$$J(w^\circ) \leq \varepsilon \tag{2-6}$$

and

$$\| y_p - y_m \| \leq \delta \tag{2-7}$$

where y_m is the model output, and $\|.\|$ denotes a norm distance between the model and the system.

In computing a neural model, the criterion function is formed by normalizing the square of errors or differences between the system output and the model output over N pairs of input-output data:

$$J = \frac{1}{N}\sum_{j=1}^{N}\frac{1}{2}(y_p(j) - y_n(j))^2 \tag{2-8}$$

ε is actually a stopping criterion for numerical iterations, and needs to be chosen properly to ensure that Equation (2-7) is satisfied.

It should be pointed out that for a given neural model set, the parameters (weights) **do not reflect any physical considerations in the unknown system**, and should be viewed just as a means for adjusting the model fit to the data observations. Therefore, an identification using a neural network approximator is essentially a **model-system output equivalence or "black box" problem.**

Chapter 3

Neural Simulator Model: Training Algorithm and Feasibility Study

In previous chapters, the basic operation of a multilayer feedforward neural network was introduced, and the place of neural networks in the general area of "identification" was formalized. It is now necessary to discuss how a neural network model of a system can be established through the training and testing procedure. In this chapter, the training algorithm based on the error backpropagation and conjugate gradient optimization method will be presented. The numerical properties of this algorithm will be demonstrated through approximating the nonlinearaties of a theoretical and a physical model. The neural network is "trained" to minimize the error between the desired output and the neural network output. Once a "user defined" acceptable error has been reached, the neural network becomes a "simulator" and must be tested for accuracy and reliability using inputs not used in the training process. Actual experimental data is used in the physical model training and testing.

3.1 Training Methods

From an optimization point of view, training a neural network is equivalent to minimizing function J in Equation (2-8). Such an error function is a multivariate function that depends on the weights in the network. The training of a neural network can be categorized as an unconstrained nonlinear optimization problem. There are a number of well known direct search methods (search for the optimal of the object function) in optimization theory which can be applied to general nonlinear large scale multivariate functions for unconstrained minimization problems [Reklatis, G. V. et al., 1983; Pike, R. W., 1986; Johansson, E. M. et al., 1992]. The search methods can be

classified into two broad categories based on whether or not the accurate values of the first derivative of the object function are required: direct search methods which use only function values, and gradient-based search methods which require the evaluation of the first derivative of the function.

The most widely used learning algorithm in neural network applications is probably the traditional back propagation method [Widrow, B. and Lear, M. A., 1990] which is derived from the simple gradient descent method. Despite its simplicity and success in supervised neural network training, this method does suffer from two major problems as experienced by the author and reported in the literature [Møller, M. F., 1993; Chen, C. H. and Lai, H., 1992; Johansson, E. M. et. al., 1992; Battiti, R. and Musulli, F., 1990]. An excessively low convergence rate exists near the solution since the modification in weights is directly related to the magnitude of the gradient, which is itself approaching zero as the iteration proceeds toward the minimum. The other problem is that instabilities arise from using an improper step size.

Among other gradient-based optimization methods, the conjugate gradient method and the quasi-Newton method have been recently introduced to static neural network applications [Johansson, E. M. et. al., 1992; Battiti, R. and Musulli, F., 1990]. The conjugate gradient method uses the first derivative of an object function to generate directions along which the object function is minimized, and performs a "line search" for a best step size at each step of the iteration. The distinct property of the quasi-Newton method is that the first derivative of the function is also used to build up an approximation to the inverse of Hessian matrix (the second derivative of the function) to mimic the high order convergence property of the pure Newton's method. Quasi-Newton type methods are computationally much more sophisticated than other gradient-based methods. One limitation associated with all gradient-based methods, when applied to a general nonlinear and nonquadratic function, is the possibility of reaching a local minimum that may not be "small enough" to meet the practical requirements, because the search for the minimum starts with an arbitrary point and always proceeds in a down

hill direction toward the nearest extremum. The question of how to obtain the location of global minimum with a general and large scale object function still remains unsolved.

Another technique that has been regarded as a potential method of reaching the global minimum in optimization problems is the Complex method originally proposed by Box [Box, M. J., 1965] for constrained optimization problems and then modified and applied to the training of neural networks [Xu, P. et. al., 1994]. The Complex method starts the search procedure with, not a single point, but a number of randomly sampled points uniformly distributed over the search space. Since the searching starts with multi-points, the chances of isolating the global minimum as the search space shrinks toward a minimum is therefore increased. In addition, the Complex method does not require evaluation of the derivatives of the object function, which makes the training algorithm simple, and easy to implement in computation, an important feature for real time applications. However, this method has difficulty coping with complex surfaces of nonlinear systems.

The question of which method can give a best overall performance for general nonlinear optimization problems still remains open; indeed, the answer to this question may be application dependent. There is, so far, no method that is significantly superior over others for all problems. It is the user's responsibility to select an effective method based on the problem at hand.

In this study, the conjugate gradient method using Polak-Ribiere formula is selected. The advantages of the conjugate method are that the conjugate gradient method has fast convergence and good stability when compared to the traditional back propagation method, and is much less complex and computationally more efficient when compared to quasi-Newton methods. In the next section, a brief introduction to the training algorithm based on the conjugate gradient method will be given, and the efficiency of this algorithm will be demonstrated through simulation and experimental identification studies on static nonlinearities.

3.2 Conjugate Gradient Algorithm in Identification of Nonlinear Relationships

3.2.1 Basis of Conjugate Gradient Methods

The theoretical concept of all conjugate direction and conjugate gradient methods is based upon the model of a quadratic function, and differs only in the way that the conjugate directions are generated. The reasons for considering a quadratic model are: (1) a general nonlinear function, as it approaches to the optimum, often behaves approximately as a quadratic, and hence can be approximated as such using a second order Taylor series expansion, and (2) any optimization technique, which is expected to succeed with general nonlinear functions, must work well on a quadratic since it is a relatively simple nonlinear function. The motivation for the algorithm is based on the following observation that if, using a linear transformation, a quadratic object function with n variables can be converted into a function that does not contain interactions between the variables, then the optimum of the function can be found by a single variable search on each of the n transformed independent variables. Thus, the one n-dimensional function minimization problem is reduced to n one-dimensional function minimization problems. The directions from the transformation are called conjugate directions. Conjugate directions can be generated by several different schemes that may or may not require the derivatives of the object function. Conjugate gradient methods use only the gradient information to generate conjugate directions.

Searching for the minimum of an object function J(.) using conjugate gradient methods can be represented by equations:

$$w^{(k+1)} = w^{(k)} + \alpha^{(k)} d^{(k)} \tag{3-1}$$

$$d^{(k+1)} = -g^{(k+1)} + \gamma^{(k)} d^{(k)} \tag{3-2}$$

where $d^{(0)} = -g^{(0)}$ and $g^{(k)} = \nabla J(w^{(k)})$. At the kth step, the variables of the object function, $w^{(k)}$ (a vector in Equation (3-1)), are modified toward the optimum along the current

conjugate directions $d^{(k)}$ at step size $\alpha^{(k)}$. The current conjugate direction vector $d^{(k)}$ is evaluated from a linear combination of the current negative gradient vector, $-g^{(k)}$, to the previous direction vector $d^{(k-1)}$, as expressed in Equation (3-2). The extension of conjugate gradient methods to general nonlinear functions has led to several different formulas [Cybenko, G., 1989; Johanson, E. M., et. al., 1992] to determine the values of $\gamma^{(k)}$ in Equation (3-2). Further computation experimental work has shown that the Polak-Ribiere formula given by Equation (3-3) seems to be favored over other formulas for the methods of the conjugate gradient types [Luenberger, D. G., 1984]:

$$\gamma^{(k)} = \frac{(g^{(k+1)} - g^{(k)})^T g^{(k+1)}}{(g^{(k)})^T g^{(k)}} \tag{3-3}$$

The best value of $\alpha^{(k)}$ at each step of the iteration is found through a line search that searches for $\alpha^{(k)}$ at which the function $J(w^{(k)} + \alpha^{(k)} d^{(k)})$ takes its minimum (note that $w^{(k)}$ and $d^{(k)}$ are known at this step, and searching for $\alpha^{(k)}$ is hence a one-dimensional function minimization problem). All single-variable search techniques from optimization theory can be applied to the line minimization.

The conjugate gradient method performs a direction search for $d^{(k)}$ and a line search for $\alpha^{(k)}$ at each step of the iteration. Theoretically, if function J is a quadratic, the iteration will terminate in, at most, M steps (ignoring round-off errors), where M is the number of function variables. When applying conjugate gradient methods to nonquadratic problems, additional direction search and line search are required and the procedure is repeated until a stopping criterion is met. For nonquadratic problems, a restart scheme is often preferred to improve the linear independence of the direction upon the previous search directions. A simple restart scheme is that the iteration procedure is interrupted every M steps and restarted by setting $d^{(k)} = -g^{(k)}$ as a restart direction. More sophisticated restart schemes are discussed by Powell [Powell, M. J. D., 1977].

3.2.2 Conjugate Gradient Training Algorithm

A challenge that still remains in neural network applications is the selection of the architecture of neural nets. The feedforward network with one or two hidden layers has been used extensively and successfully in many applications of nonlinear approximations. In this study, a two hidden-layer feedforward neural network is selected to explore its capabilities of nonlinear approximations for a real hydraulic component. Each neuron at hidden layers performs a summation and a nonlinear mapping by a sigmoidal function, and has a bias associated with it. A neuron at the output layer only sums the weighted inputs.

With reference to Figure 3.1, the output of the network at jth node $y_n(j)$ depends on the weights as well as the inputs to the network, and is given by:

$$y_n(j) = \sum_{\phi=0}^{L_3} v_2(\phi) \cdot w_{34}(\phi, j) \qquad (3\text{-}4)$$

$$j=1,2, \ldots, L_4$$

where $v_2(0)$ and $w_{34}(0,j)$ form a bias for the jth neuron at the output layer, and $v_2(0)$ is set to zero, since the network used in this study does not have biases at its output layer.

$v_2(\phi)$ in Equation (3-4) is the output of ϕth nonlinear function at the second hidden layer:

$$v_2(\phi) = \Phi[s_2(\phi)] = \tanh(s_2(\phi)) \qquad (3\text{-}5)$$

where

$$s_2(\phi) = \sum_{\phi_1=0}^{L_2} v_1(\phi_1) \cdot w_{23}(\phi_1, \phi) \qquad (3\text{-}6)$$

$v_1(0) = 1$ and $w_{23}(0, \phi)$ form a bias for ϕth neuron at the second hidden layer. $v_1(\phi_1)$ is an output of the nonlinear function at ϕ_1th node of the first hidden layer and is given by:

$$v_1(\phi_1) = \tanh(s_1(\phi_1)) \qquad (3\text{-}7)$$

where

$$s_1(\phi_1) = \sum_{\phi_2=0}^{L_1} u(\phi_2) \cdot w_{12}(\phi_2, \phi_1) \tag{3-8}$$

where $u(0) = 1$ and $w_{12}(0, \phi_1)$ form a bias for ϕ_1th neuron at the first hidden layer.

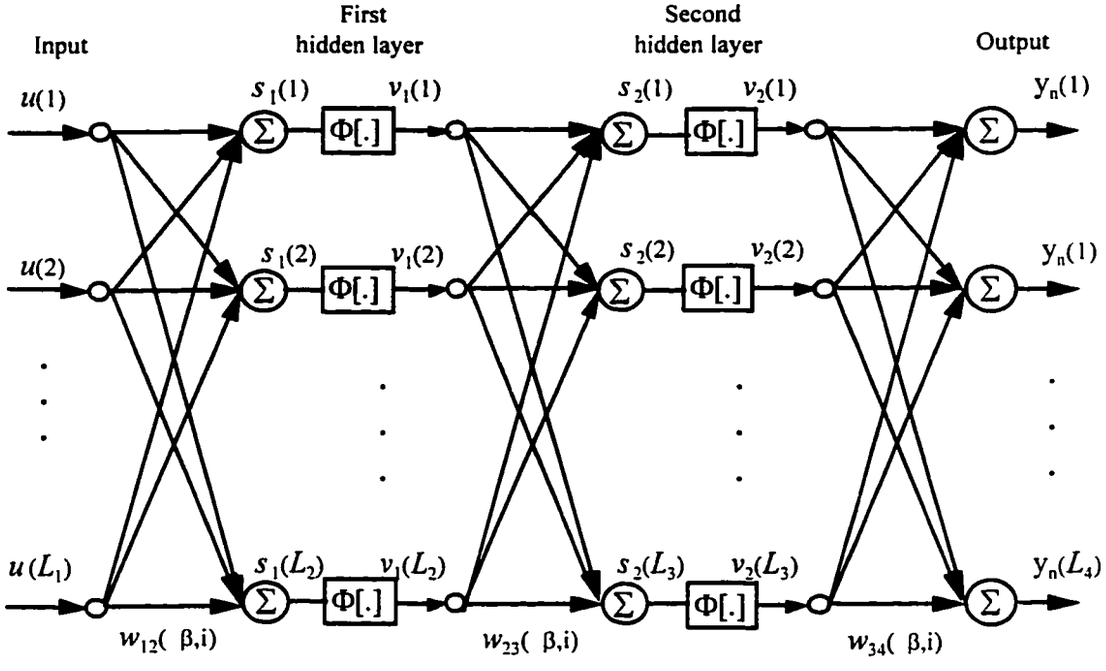


Figure 3.1 The neural network used for this study

The object function J for training the neural network shown in Figure 3.1 is formed by normalizing errors or differences between the desired output y_p and the neural net output y_n over N pairs of input-output data:

$$J = \frac{1}{N} \sum_{q=1}^{N} \sum_{j=1}^{L_4} \frac{1}{2} \cdot (y_p(j) - y_n(j))^2 \tag{3-9}$$

Training the network is to adjust all weights (denoted as $w_{12}(\beta,i)$, $w_{23}(\beta,i)$ and $w_{34}(\beta,i)$ in Figure (3.1)) according to Equations (3-1), (3-2) and (3-3) so that J takes on its minimum value. The gradient vector $g^{(k)} = \nabla J(w^{(k)})$ appears in Equations (3-2) and (3-3) is obtained from Equation (3-9):

$$\nabla J(w^{(k)}) = -\frac{1}{N}\sum_{q=1}^{N}\sum_{j=1}^{L_3}(y_p(j) - y_n(j)) \cdot \frac{\partial y_n(j)}{\partial w^{(k)}} \tag{3-10}$$

The gradients of $y_n(j)$ with respect to the weight vector $w^{(k)}$ in Equation (3-10) can then be derived backward from Equations (3-4), (3-5), (3-6), (3-7) and (3-8) as follows:

(1) the gradients of $y_n(j)$ with respect to $w_{34}(\beta,i)$:

$$\frac{\partial y_n(j)}{\partial w_{34}(\beta,i)} = v_2(\beta) \cdot \delta_{ij} \tag{3-11}$$

(2) the gradients of $y_n(j)$ with respect to $w_{23}(\beta,i)$:

$$\frac{\partial y_n(j)}{\partial w_{23}(\beta,i)} = \sum_{\phi=1}^{L_2} w_{34}(\phi, j) \cdot \frac{\partial v_2(\phi)}{\partial w_{23}(\beta,i)} \tag{3-12}$$

$$\frac{\partial v_2(\phi)}{\partial w_{23}(\beta,i)} = \Phi'[s_2(\phi)] \cdot \frac{\partial s_2(\phi)}{\partial w_{23}(\beta,i)} \tag{3-13}$$

where $\Phi'[s_2(\phi)] = 1 - \tanh^2(s_2(\phi))$.

$$\frac{\partial s_2(\phi)}{\partial w_{23}(\beta,i)} = v_1(\beta) \cdot \delta_{i\phi} \tag{3-14}$$

(3) the gradients of $y_n(j)$ with respect to $w_{12}(\beta,i)$:

$$\frac{\partial y_n(j)}{\partial w_{12}(\beta,i)} = \sum_{\phi=1}^{L_2} w_{34}(\phi, j) \cdot \frac{\partial v_2(\phi)}{\partial w_{12}(\beta,i)} \tag{3-15}$$

$$\frac{\partial v_2(\phi)}{\partial w_{12}(\beta,i)} = \Phi'[s_2(\phi)] \cdot \frac{\partial s_2(\phi)}{\partial w_{12}(\beta,i)} \tag{3-16}$$

$$\frac{\partial s_2(\phi)}{\partial w_{12}(\beta,i)} = \sum_{\phi_1}^{L_1} w_{23}(\phi_1, \phi) \cdot \frac{\partial v_1(\phi_1)}{\partial w_{12}(\beta,i)} \tag{3-17}$$

$$\frac{\partial v_1(\phi_1)}{\partial w_{12}(\beta,i)} = \Phi'[s_1(\phi_1)] \cdot \frac{\partial s_1(\phi_1)}{\partial w_{12}(\beta,i)} \tag{3-18}$$

$$\frac{\partial s_1(\phi_1)}{\partial w_{12}(\beta,i)} = u(\beta) \cdot \delta_{i\phi_1} \tag{3-19}$$

where $\Phi'[s_1(\phi_1)] = 1 - \tanh^2(s_1(\phi_1))$.

In implementing the conjugate gradient algorithm presented above, the line search for the best value of step size $\alpha^{(k)}$ uses a parabolic fit to bracket (enclose) a

minimum of the error function and then a golden section search technique [Pike, R. W., 1986] to isolate the minimum. The simple restart scheme was also adopted. A complete sequence of the training requires a number of iteration steps greater than the number of the network weights, because of the non-quadratic form of the error function and round-off errors in computation. The major steps of this algorithm are briefly summarized as follows:

Initialization:

Initialize weight vector $\mathbf{w}^{(0)}$;

Compute $J^{(0)}$ and $\nabla J^{(0)}$.

Step 0: Set initial direction vector $d^{(0)} = -\nabla J^{(0)}$.

Iteration:

Step 1: Determine $\alpha^{(k)}$ from a line search;

Update the weight vector to its new point $w^{(k+1)}$ using Equation (3-1).

Step 2: Evaluate $J^{(k+1)}$ and $\nabla J^{(k+1)}$;

Check stopping criteria, and terminate if the criterion is met.

Step 3: If $k+1$ is equal to M (the number of weights), set $k = 0$. Go to Step 0 and restart the iteration; otherwise go to Step 4.

Step 4: Compute new direction vector $d^{(k+1)}$ using the Polak-Ribiere formula in Equation (3-3).

Step 5: Set $k = k+1$. Go to Step 1.

The procedure is terminated when one of the following stopping criteria is met:

(1) the change in error function J is small enough to meet:

$$\tfrac{1}{2}(|J^{(k+1)} - J^{(k)}|) < 10^{-7}(|J^{(k+1)}| + |J^{(k)}|) \qquad (3\text{-}20)$$

(2) the gradient of the error function is approximately zero, i.e.:

$$(g^{(k)})^T g^{(k)} < 10^{-10} \tag{3-21}$$

(3) the iteration arrives at its maximum iteration number.

The numerical values, 10^{-7} in Equation (3-20) and 10^{-10} in Equation (3-21) can be specified differently for different accuracy required.

3.3 Illustration of the Conjugate Gradient Training Algorithm Using a Theoretical Model

A theoretical model of a square root relationship for a variable flow orifice is first considered to implement the training algorithm. The model is expressed by:

$$Q = X\sqrt{P} \tag{3-22}$$

Q, X and P are normalized flow rate, orifice opening and pressure drop across the orifice. The model has two inputs, X and P, and one output Q, which corresponds to two inputs $u(1)$ and $u(2)$, and one output $y_n(1)$ of its neural network identifier. Nine neurons and 5 neurons were selected for the first and the second hidden layer, respectively. Indeed, several different numbers of neurons at the two hidden layers have been experimented in the training. The results showed that the network with nine neurons at the first hidden layer and five neurons at the second hidden layer gave a better overall accuracy, and this morphology was therefore adopted in this study.

3.3.1 Training

The training data set consisted of 21 points of X and 21 points of P with both groups of data evenly distributed over the region [0,1]. For a given point of X, there were 21 values of Q that corresponded to 21 points of P. Therefore, the total number of the data pairs used for the training was 21 x 21 = 441. The program was executed on a PC 486 33 MHz microcomputer. The training took about 5 hours on this computer and was terminated with the error function (defined by Equation (3-9) where N = 441) at 9.142×10^{-6}. It was observed that the error function decreased asymptotically toward its

minimum during the training. Indeed, the error function took only about 25 minutes to be reduced to a small value of 3.87×10^{-5} and then took the major part of the training time to refine this value.

3.3.2 Testing

Other 26 by 26 data pairs were sampled over the region [0,1] from the theoretical model of Equation (3-22) and fed into the trained network for the model testing. Twenty-six tests were performed with each corresponding to a fixed value of X and 26 values of P. The result for each testing is presented using root mean square (RMS) values ($\text{RMS} = \sqrt{\frac{1}{N}\sum_{q=1}^{N}(y_P(1) - y_n(1))^2}$) and maximum differences (MDIF) between the desired outputs $y_p(1)$ and actual neural net outputs $y_n(1)$ over each test. Maximum RMS and maximum MDIF values over 26 tests are 0.00902 and 0.03915, respectively, which occur at the 5th test as shown in Figure 3.2 where only 5 test results are presented graphically for demonstration purposes. All 26 test results consistently show that there is an excellent agreement between the outputs from the neural network identifier and from the theoretical model.

Several conclusions can be made from the above results: (1) the conjugate gradient training algorithm provides a fast convergence rate; (2) the selected structure of the neural network is able to obtain quite satisfactory results for identifying a theoretical square root relationship, and (3) for the selected neural network and the nonlinear model, the training data information is sufficient to achieve excellent generalization properties.

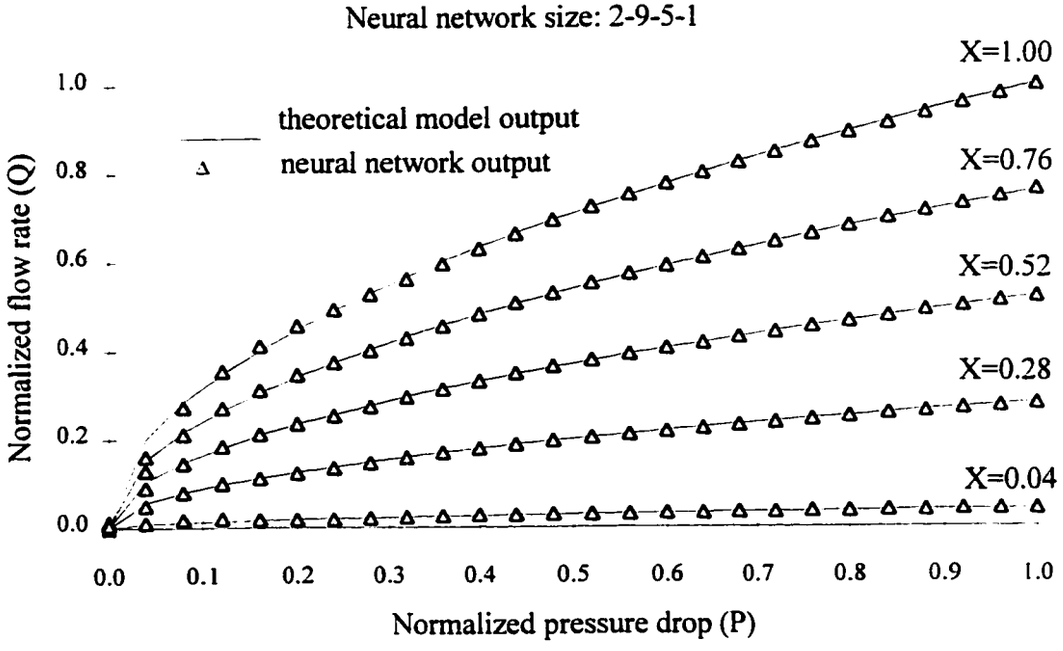


Figure 3.2 Testing results using a theoretical model

3.4 Implementation of Conjugate Gradient Training Algorithm Using Experimental Data

3.4.1 Experimental System and Data Acquisition

The block diagram of the experimental system for measuring flow rate as a function of orifice opening and pressure drop is shown in Figure 3.3. The valve to be tested was a Parker F800S flow control valve. The supply pressure and flow rate in this experiment were 2800 psi and 15 GPM, respectively. The pressure drop across the test valve was adjusted by another flow valve mounted upstream of the test valve. The opening of the test valve was measured by a dial gage attached to the valve spool. The pressure drop and flow rate were measured in the form of digital readings from electrical signal amplifiers.

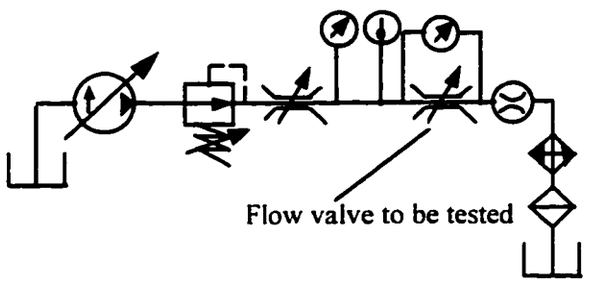


Figure 3.3 Block diagram of the experimental system for data measurement

In each test, for a given X (the spool displacement of the test valve), the flow rate Q was collected for both increasing and decreasing P for two complete cycles. There were nine tests carried out. The maximum value of pressure drop was 2700 psi, and the maximum spool displacement was 0.184 in. The temperature during the experiment was observed to be 45°C ±4°C.

It was difficult to obtain an accurate measurement at the very low flow region especially for very small orifice openings. The training range for which the identification was performed did not, therefore, cover the area near the origin. Also the data records were influenced by measurement noise and other factors, such as temperature drift.

A change in the sensitivity of the flow rate Q to the spool displacement X can be observed as a transient region nearing $X > 0.147$ in (normalized to 0.799), as illustrated in Figure 3.4. This indicates that the theoretical linear relationship normally assumed between the flow rate and the displacement of the valve spool for a given pressure drop is not maintained over the full range. This nonlinear sensitivity arises from the internal structure of the valve.

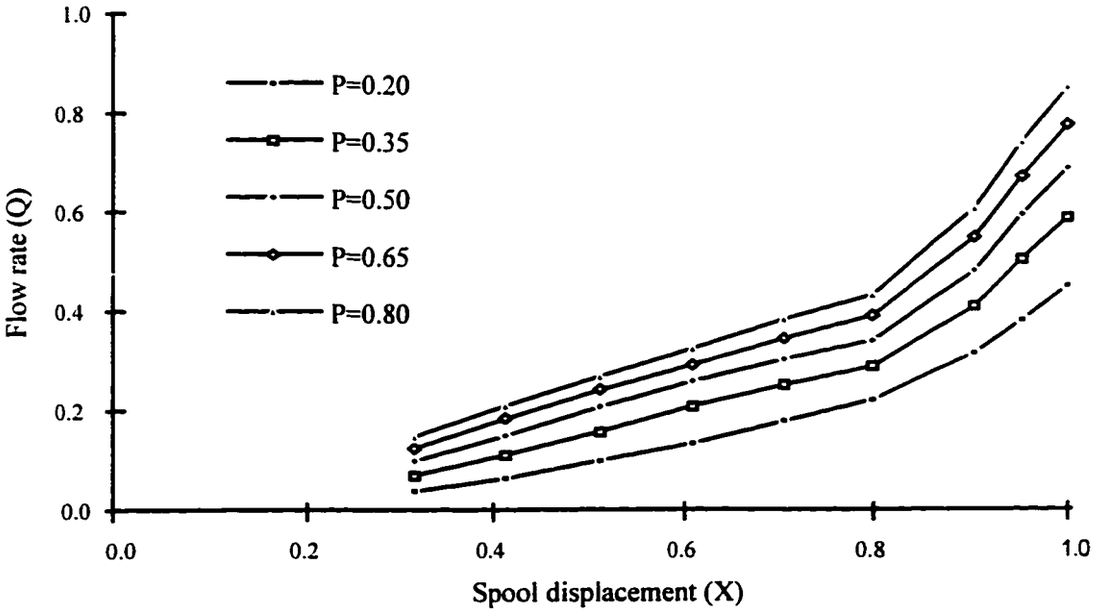


Figure 3.4 Sensitivity of flow rate to spool displacement (Q , X and P are normalized)

Since the training information must be sufficient to fully represent the nonlinear range of the component for which the neural net was trained for, the data from eight tests were selected for the training, and the other data points from one test were used for the final testing of the trained neural identifier. Four different cases were examined with each having a different selection of data groups for training and testing in order to observe the neural network's identification abilities and the accuracy of the final results over different selections of data ranges. All training and testing data were normalized by the maximum readings for flow rate, pressure drop and the displacement of the spool, respectively.

3.4.2 Training Results

The examination of only two cases was presented as follows:

Case 1: the data from the test at X = 0.511 (normalized) were used for testing and the remaining data were for training.

Case 2: the data from the test at X = 0.951 (normalized) were for testing and the remaining data for training.

In some cases, more than one execution of the training program was necessary to obtain satisfactory accuracy of the results. Normally, well terminated training would give nearly identical results in the same case even though they began with different initial weights and did not have exactly the same final weights. However, a normally terminated training may not necessarily give an acceptable training result if the solution was trapped into a local minimum that was not "small enough" to meet the required accuracy. Some of the trainings experienced slight difficulties in the transition region. The training over the lower flow rate range showed to be a lot easier than over the transition and higher flow rate ranges.

For different runs of training, all weights were initialized differently with random numbers uniformly distributed over the range from -1 to +1. The training took several hours on a PC 486 33 MHz microcomputer, and iterations terminated when one of the three criteria was met. The error function was monotonically and quickly reduced to an order of 10^{-4} in less than 25 minutes before the restart scheme was resumed, and then refined to an order of 10^{-5} toward the minimum of the error function with the rest of training hours. Training results for two cases are summarized in Table 3.1. A typical training result for Case 1 is presented graphically in Figure 3.5.

Table 3.1 Training results for a flow valve using experimental data

Case No.	Final error function values	No. of iteration steps	Training time
1	5.719×10^{-5}	1352	6 hrs. 11 min
2	6.976×10^{-5}	1092	5 hrs. 53 min

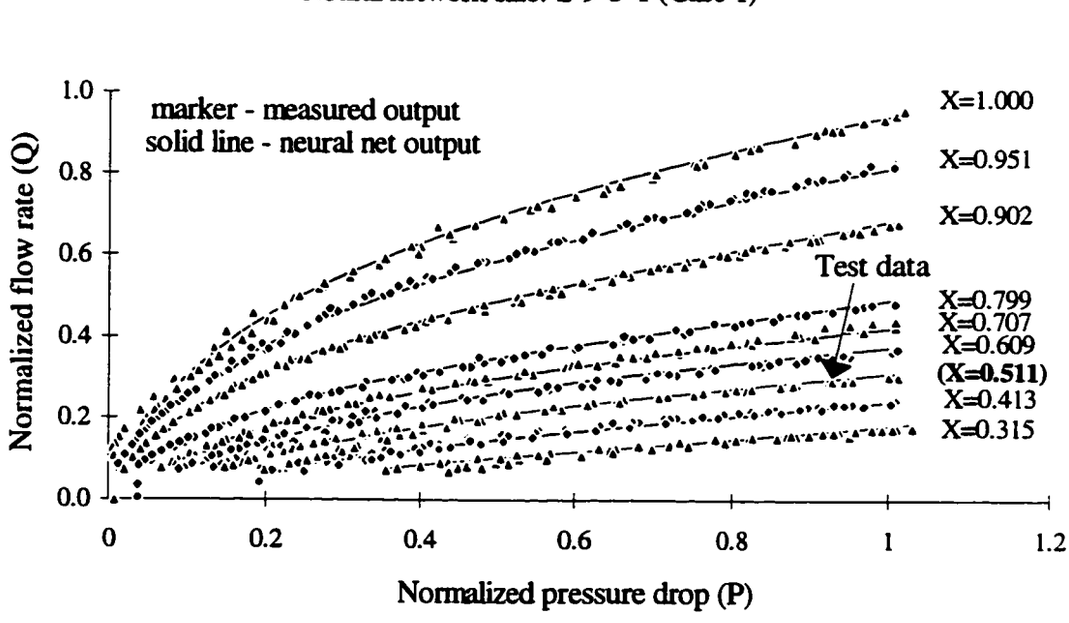


Figure 3.5 Training and testing results using experimental data for Case 1

All results show consistently that the training of the neural network has driven the network to a best fit of the experimental data even for the low flow region where the data are somewhat scattered. An excellent agreement between the neural network outputs and the experimental output has been achieved through training. The training results show that the neural network is able to reproduce the experimental data with satisfactory accuracy.

3.4.3 Testing Results

Testing was then performed using data not used for the training. The testing result for each case is plotted and overlaid on the plot of the corresponding training result, as shown in Figures 3.5 and 3.6, for ease of observation. As can be seen in these figures, two testing results present different accuracy of the interpolations though two trainings have been quite successful.

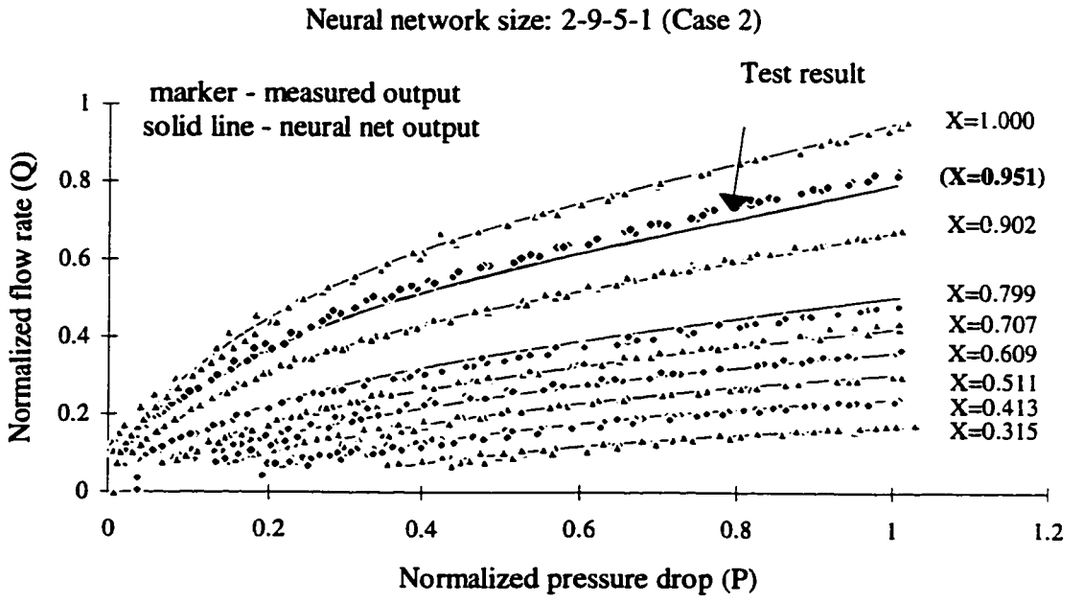


Figure 3.6 Training and testing results using experimental data for Case 2

The testing results in Case 1 show excellent abilities of the trained neural network to interpolate. The training information used in Case 1 is hence considered to be sufficient and adequate over the nonlinear region nearing $X = 0.511$, and the neural network is able to realize the identification of the nonlinearities for these particular ranges.

The testing results from Case 2 present somewhat insufficient accuracy of the interpolations. The cause for this phenomenon is considered to be insufficient training information over the selected testing ranges. In Case 2, the training data that does not include the points at $X = 0.951$ appears too distant in terms of X values, as can be seen in Figure 3.6, and the interpolation accuracy is reduced especially at the high pressure region.

The above results show that the selection of training data is crucial, because it has a significant effect on the final accuracy, and hence the quality of the identification of the nonlinearities. For a complete identification procedure, data for both training and testing must be sufficient in the number of points and be properly distributed over the full range of nonlinear relationships. Sufficient training data can realize the neural network's capabilities of reproducing and interpolating with high accuracy. Sufficient testing data can effectively assess the quality of the neural identifier. However, it should be pointed out that when using experimental data to train a neural network, the accuracy may not be necessarily achieved to any high degree, because of realistic constraints on the measurement of experimental data, such as instrumentation noise or limited resolution of measurement and other factors.

3.5 Discussion

The conjugate gradient method has been shown to be an excellent technique for training a neural net to a theoretical nonlinear function. The ability of the trained neural network to generalize has also been established. Convergence is relatively rapid and the solution is stable.

The technique has also been shown to be appropriate for the identification of a typical hydraulic component, an orifice. In this case, noisy experimental data have been shown to be a viable training set. Once more the ability of the neural network to generalize has been demonstrated by its approximation of a set of experimental data separated from those used in the training.

By changing the sets of training and testing data, it has been shown that the accuracy of the neural net's interpolation is dependent on the "coverage" of the training data. Clearly, the appropriate selection of training data is extremely important to the accurate identification of a hydraulic component.

Chapter 4

Simulation Studies on the Identification of Dynamic Systems

In Chapter 3, the rapid convergence and stable solution capabilities of the conjugate gradient algorithm used in this research was established. It is now necessary to investigate the feasibility of using a dynamic neural net model and the conjugate gradient algorithm to identify a nonlinear dynamic system. The investigation will be carried out through two stages: simulation study and experimental identification.

The simulation study will focus on the examination of the modeling error behavior using perfect data. The idea is that since the modeling error observed under noise-free conditions (to be defined as model structure error in the next section) is caused by the virtual differences in model structure and plant[1] structure, it reflects the ultimate approximation capability of the neural model for the given plant. In general, a model calculated using noisy data may not give satisfactory performance if it can not achieve the desired modeling accuracy under noise-free condition. The experimental identification will focus on the realization of a practical identification of a physical hydraulic component.

This Chapter will present three typical simulation examples of identifying nonlinear dynamic plants using a partially recurrent type of neural network. The structure of dynamic neural net models and the training schemes are first described, and then off-line batch training identification is performed with noise-free data generated

[1] Because of the common use of the term "plant" in the literature, this term used throughout this thesis denotes a real-world system or a mathematical model of a real-world system. In general, the term "plant" is referred to as a process that produces desired outputs for the identifications.

from mathematical expressions of the plants. The model structure errors will be examined from the training and testing results. A further error analysis will be considered in the next chapter (Chapter 5).

4.1 Model Structure Error

The residual term $e_r(k)$ in Equation (2-3) is defined as fitting error or modeling error. The modeling error consists of model structure error $e_s(t)$, caused by the structure differences between the plant and its model, and the experimental data noise $e_m(t)$. In practical identification, $e_s(t)$ and $e_m(t)$ are not separable. The structure error can be imagined as the differences between the model output and the plant output observed under the ideal condition of noise-free (input and output data are not contaminated by measurement or plant noise). A desirable mathematical property of a model structure is such that the structure error can be minimized to an arbitrarily small value (i.e. the model accuracy can achieve an arbitrary precision).

The static neural net models with mulitlayer feedforward structure have been theoretically proven to be able to approximate any continuous functions to any degree of accuracy, provided enough hidden units are available [Hornik, K., Stinchcombe, M. and White, H., 1989], and have been referred to as a "universal approximator". The possession of this property implies that mathematically, structure error should tend approximately to zero. In real implementations with perfect data observations, however, some factors, such as the finite order of the input and output variables in the model structures, the finite number of neurons at the hidden layers, complexity of input-output relationship of the specific plant, and finite precision in computers, etc. will certainly put practical constraints on the achievable modeling accuracy. That is, in real situations, the structure error can never tend to zero. In this regard, the property of approximation to any degree of accuracy cannot be applied to real-word problems, and the property of having "small enough" structure errors in a function approximation should be imposed instead.

In using a partially recurrent neural network for dynamic system identification, an important aspect is that the modeling error behavior is no longer similar to that in static nonlinearity identification. The error is characterized by its recurrent accumulation through model feedback paths that are introduced for the model to capture the plant dynamic features.

The significance of studying the structure error behavior using a recurrent type of neural models is to reveal the mechanism of how the structure errors are accumulated, and to explore the capabilities and limitations of this recurrent type of neural models in approximating a nonlinear dynamic plant.

4.2 Basic Dynamic Neural Network Structure

There are various structures of dynamic neural networks that have been found in many applications of identification and prediction for control purposes [Hush, D. R. and Horne, B. G., 1993; Narendra, K. S., 1990; Jin, L. et. al., 1994, 1996]. One of the more popular neural network model structures to approximate the dynamics of a nonlinear plant is formed by adding external feedback paths to a multi-layered feedforward net, and is referred to as a partially recurrent neural network. Figure 4.1 shows a block diagram of such a dynamic neural network in which its dynamic features are formed by using a group of time delay units on the inputs and output feedback of the net, respectively.

By properly choosing the morphology of the multilayer feedforward network and the number of the time delay units, a class of dynamic neural models is constructed and expressed by:

$$y_n(k) = N_m(u(k), u(k-1), u(k-2), ..., u(k-m), y_n(k-1), y_n(k-2), ..., y_n(k-n)) \qquad (4\text{-}1)$$

Note that as the model, after estimated, will be used in simulation, i.e. independently of the plant, the previous model outputs $y_n(.)$, not plant outputs $y_p(.)$, are used in the calculation of the current model output.

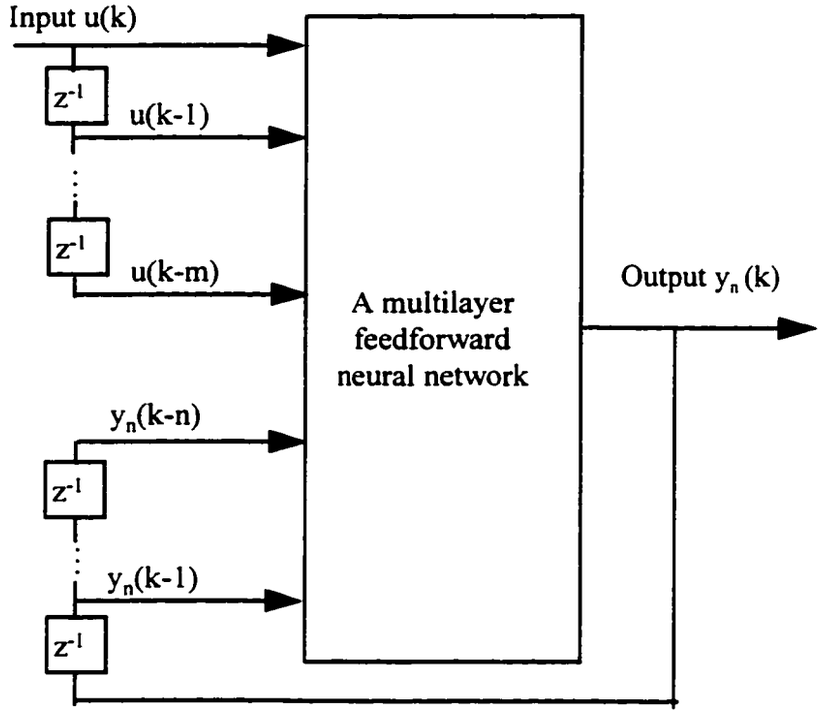


Figure 4.1 A neural dynamic simulation model

4.3 Nonlinear Regression Schemes of NNARX and NNOE

An "estimate" of a dynamic neural model is to fit the model, N_m, to a series of input-output data so that the resulting model can be a best approximation of the plant as the fitting error is minimized. This estimation problem can be viewed as a nonlinear regression problem, and the input-output data series is referred to as a nonlinear regressor, denoted as $\Psi(k)$.

There are two regression schemes that have been applied for the implementation of dynamic neural model estimates: NNARX (Neural Network based AutoRegressive

eXogenous (inputs)) and NNOE (Neural Network based Output Error) [Sjoberg, J., 1993]. NNARX and NNOE are named by their linear counterparts ARX and OE used in [Soderstrom, T. and Stoica, P., 1989], and both also correspond to series-parallel and parallel models used in [Narendra, K. S., and Parthasarathy, K., 1989, 1990]. The difference between the two schemes is in the regressor, $\Psi(k)$. In the NNARX scheme, the past **plant** outputs, $y_p(k-1)$, $y_p(k-2)$, ... are used in the regressor:

$$\Psi(k)=\{u(k), u(k-1), u(k-2), ..., y_p(k-1), y_p(k-2), ...\} \quad (4\text{-}2)$$

In the NNOE scheme, the past output in the regressor is from previous **model** outputs which are in turn the function of currently estimated model parameters or the net weights. The regressor for NNOE is therefore a pseudo regressor:

$$\Psi(k,w)=\{u(k), u(k-1), u(k-2), ..., y_n(k-1,w), y_n(k-2,w), ...\} \quad (4\text{-}3)$$

The block diagram of NNARX and NNOE schemes are shown in Figure 4.2.

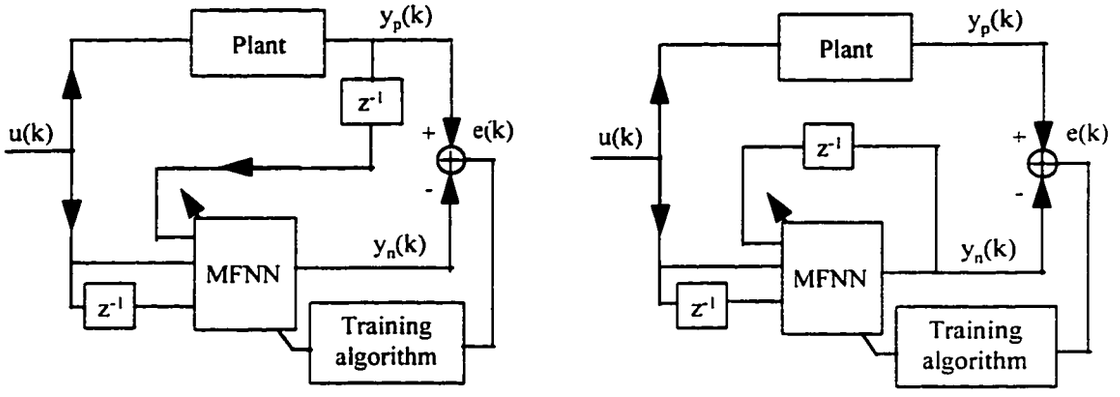


Figure 4.2(a) NNARX regression scheme, (b) NNOE regression scheme

Ideally, both schemes would be considered equivalent when $y_n(k)$ approaches to $y_p(k)$ as $e_f(k)$ approaches to zero. Indeed, if the estimates in the both schemes converge equally well, then NNOE and NNARX make no difference in terms of accuracy of the final results. The convergence properties in using NNARX and NNOE to estimate model parameters in off-line batch training have been investigated through extensive

computing experiments by the author. It has been found that the NNARX scheme has some numerical advantages over the NNOE scheme. The NNARX is less complex in algorithm, has superior convergence rate and always reaches a stable solution. The NNOE, on the other hand, may encounter difficulties in the convergence of the solution. One of the reasons is that the NNOE makes use of past model outputs evaluated from currently estimated parameters, and, in batch training, the error summed over a batch grows rapidly before one update of the parameters can be made to reduce the error function effectively. The other reason is that the calculation of the gradient of the cost function involves a summation over the derivatives of current model outputs and all past model outputs, which could be an excessive value at the initial estimate stage. This convergence problem in NNOE has been also stressed by Narendra and Parthasarathy in their report [Narendra, K. S. and Parthasarathy, K., 1989].

It was decided to use the NNARX scheme in the initial training process in this study as it always gave asymptotically stable solutions. After a satisfactory convergence has been achieved and the error has been minimized, the simulation model is obtained by switching the output feedback path from the plant output to the model output. The simulation model is also referred to as a "model-driven" mode in contrast with a "plant-driven" model that still uses plant outputs feedback to the net.

4.4 Input Signal Requirements and Model Validation

4.4.1 "Richness" Input Requirement

An important requirement for input signals is that they must excite all modes of the plant in order to obtain full information about input-output properties of that plant. Such an input signal has been referred as to "persistently exciting" or "general enough" or "rich enough" in the literature. For the identification of a dynamic system, the requirement of persistent excitation means that the input must be sufficiently rich in frequency content and in amplitude variations.

Some typical inputs used in dynamic system identification are pseudo binary signals or random signals that are (approximately) "white" characteristics in the frequency band. These white signals have been traditionally considered as "optimal" since the energy they contained is distributed over the full frequency spectrum with about equal power. In practical situations, however, the frequency band, over which a physical plant can be excited to have meaningful responses, is often limited to a very low frequency range. This requires that the input signal should contain the energy concentrated at the low frequencies that cover the limited bandwidth of the plant. Direct use of white signals would give an unfair and unnecessary advantage to higher frequencies, since lower frequencies in a higher intensity is desirable. Low pass filtering will be used to modify the white signals to an input that has a proper frequency bandwidth with respect to the plant's bandwidth. Amplitude of the input should be specified properly at the same time to ensure that the input covers all possible amplitudes when the model is in practical use.

4.4.2 Model Validity Tests

Validity tests must be performed to assess whether or not the estimated model is adequate for the intended purpose. Based on the approximation approach that has been formalized in Chapter 2, the validity criteria are defined in forms of Equations (2-6) and (2-7), where the parameters ε and δ will reflect the requirements on the model quality for a particular identification problem. Their actual values are problem dependent and often are quite subjective. The desirable properties of a model are its "generalization" capability and sufficient accuracy. Since a model is to be used as a simulator of a real plant, the best test method is simulation using "new" input data — data that are not used during the estimation. To secure reasonable test results, several entirely different types of inputs with variable amplitudes should be used, and sufficient number of tests with different inputs should be performed. Suggested types of inputs for validity tests are different step inputs, sinusoidal waves, and random sequences.

In presenting testing results, normalized RMS values of testing errors (defined by RMS divided by the maximum plant output), denoted as NRMS, and graphic plots of testing results will be used together to provide a complete picture on how well the model agrees with the observed data from the plant. NRMS will give a quantitative description of modeling error on average, while plots will provide a graphic description of modeling accuracy from which local modeling errors over each testing data pair can be observed.

4.5 Simulation Examples of Nonlinear Dynamic Plant Identifications

The following presents three examples of identifying nonlinear dynamic plants simulated by causal discrete system equations assuming time-invariant systems. The neural model for each plant was estimated using the NNARX scheme. All weights of the model were initialized with random numbers uniformly distributed over the range from -0.25 to +0.25, and then were iteratively modified to minimize the error function. The training process was terminated when one of the three stopping criteria was met:

(1) the gradient of the error function approximated to zero, or

(2) the error function no longer decreased, or

(3) the pre-specified maximum iteration number was exceeded.

The training program was executed on a PC 486-50 Hz microcomputer. The identification results are shown via training results. The validity testing results are obtained using a new batch of data not used in the training process. The results from identifying different plants are compared to each other in order to observe the effects of plant "apparent dynamics"[2] on the modeling accuracy.

Plant 1: The first plant considered is an underdamped system having a damped frequency of about 5 Hz. With a sampling frequency of 400 Hz, the plant can be approximated by the difference equation:

[2] In this thesis, "apparent dynamics" refers to the observed performance of the plant which is a result of sampling speed and the continuous system dynamics.

$$y_p(k) = \frac{2.05y_p(k-1) + 0.00625y_p{}^2(k-1) - y_p(k-2) + 0.003125u(k)}{1.05 + 0.0125|y_p(k-1)|} \quad (4\text{-}4)$$

The inputs used to identify the plant are filtered random signals that contain frequencies up to 20 Hz with magnitudes nearly uniformly distributed between 0.0 and 1.0. A spectrum plot shows that the input bandwidth of 20 Hz is wide enough to completely cover the frequency range over which the plant can be excited.

It has been found that in the training for this particular plant, a neural net with one nonlinear hidden layer converges quicker than a net with two nonlinear hidden layers for the same training accuracy. The selected neural structure has 3 input nodes (one is input, the other two are delayed outputs: y_p(k-1), y_p(k-2)) and 15 neurons at the nonlinear hidden layer.

During the training, the error function was observed to be monotonically and rapidly reduced to the order of $3.0\text{x}10^{-5}$ in less than 9 minutes as the first two batches of data were fed (one batch data consists of 1000 input-output pairs). The training then took several hours (using 15-30 batches of data) to refine the error function to an order of $5.0\text{x}10^{-8}$. Trainings could also concentrate on the dynamic characteristics at lower frequencies by adjusting the bandwidth of the input signal filter.

Training results:

One typical training result for last batch of training data is shown in Figure 4.3. For clarity, 400 of 1000 data pairs are plotted. The NRMS value over 1000 data points is also marked on the plot. The result shows that for the training data, the model outputs are able to follow the plant outputs with high accuracy, and the corresponding curves on the plot cannot be distinguished when superimposed.

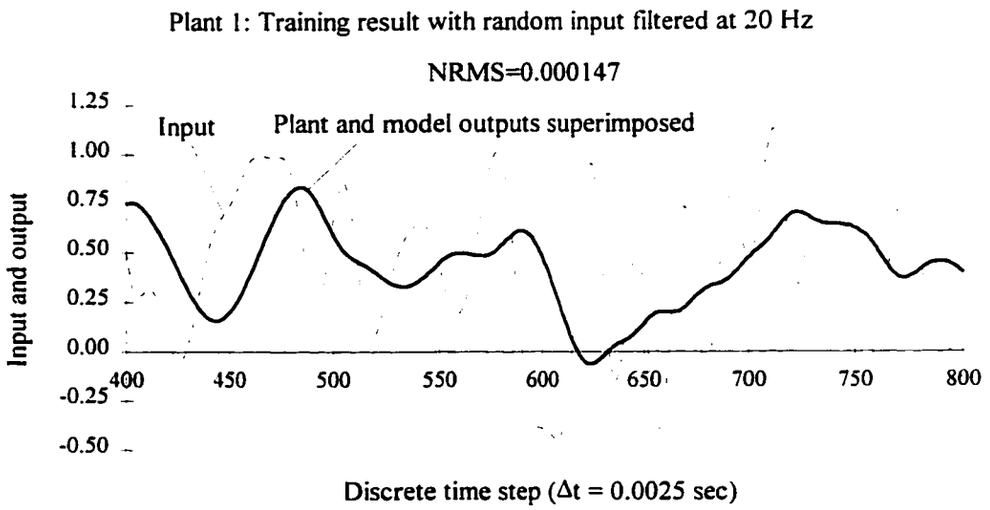


Figure 4.3 Training result for the under-damped 2nd order nonlinear dynamic plant

Testing results:

With all weights fixed, the model was subjected to fresh data to test both its generalization property and its modeling accuracy. A number of tests were carried out in both the plant-driven and model-driven modes using different types of input series. The typical testing results using three different types of inputs ((a) filtered random signals with a bandwidth of 5 Hz; (b) step inputs; (c) one Hz sinusoidal waves) are presented in Figures 4.4(a), 4.4(b), 4.4(c) for plant-driven mode and in Figures 4.5(a), 4.5(b), 4.5(c) for the model-driven mode, respectively. The corresponding NRMS value over each batch (1000 data pairs) is also marked on each plot.

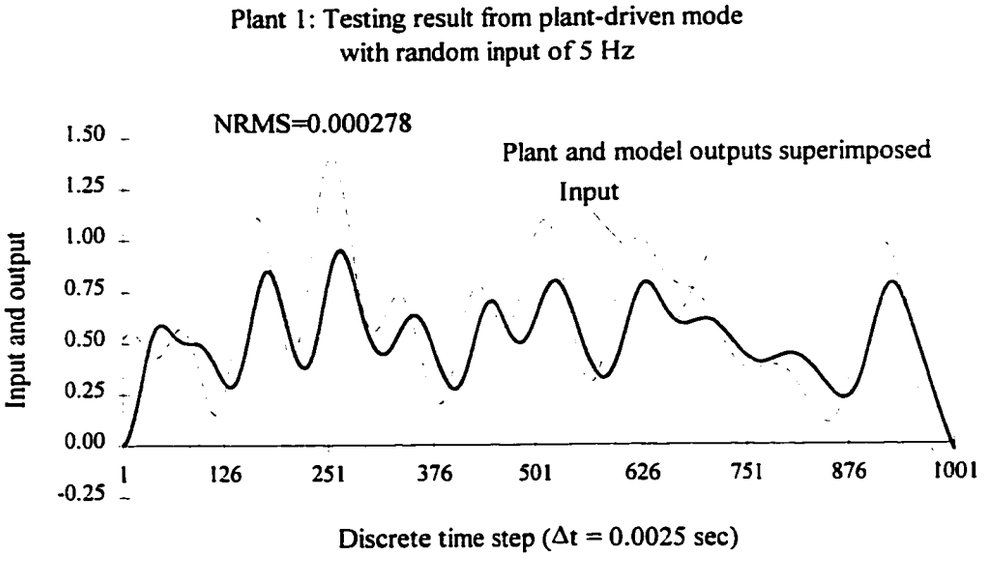


Figure 4.4(a) Plant-driven testing result using random input for Plant 1

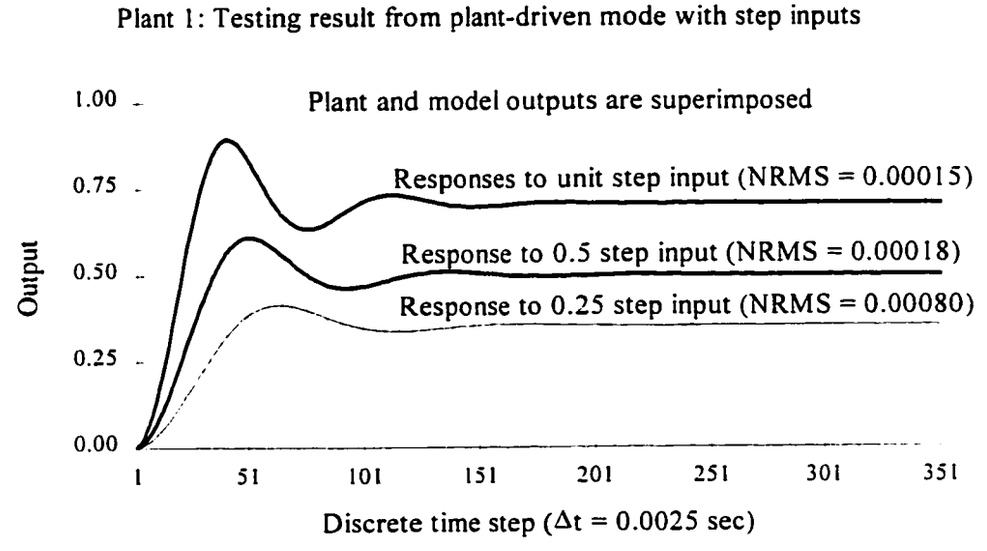


Figure 4.4(b) Plant-driven testing result using step inputs for Plant 1

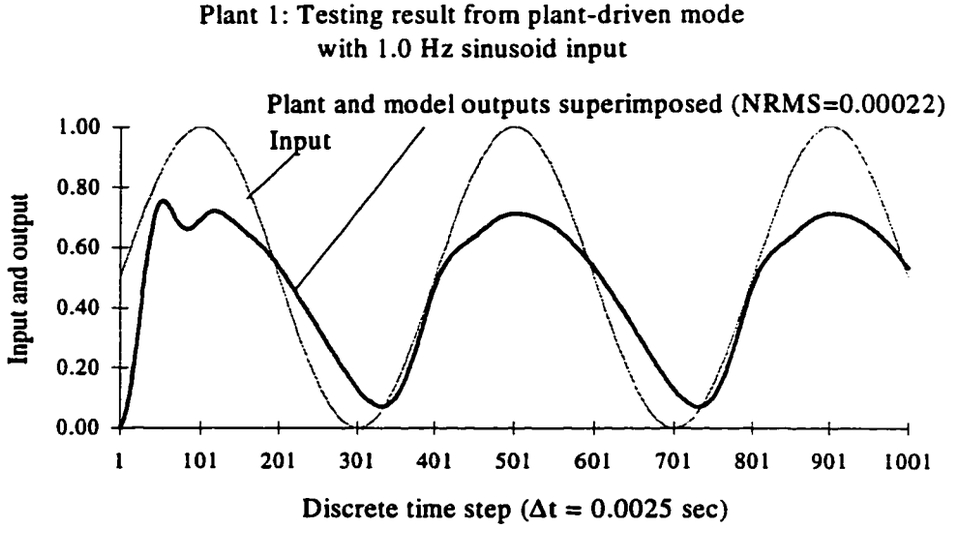


Figure 4.4(c) Plant-driven testing result using sinusoidal input for Plant 1

Figure 4.4 Plant-driven testing results for plant 1

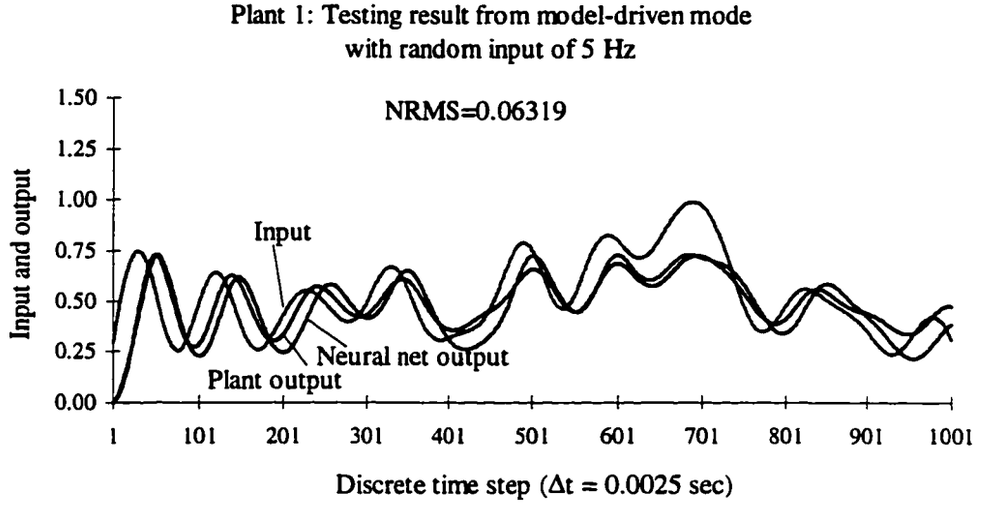


Figure 4.5(a) Model-driven testing result using random input for Plant 1

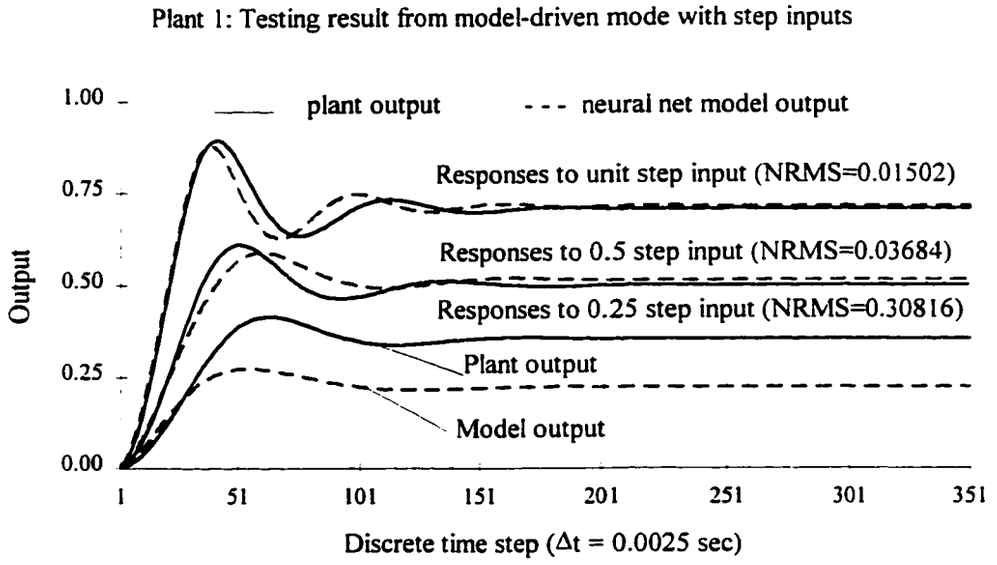


Figure 4.5(b) Model-driven testing result using step inputs for Plant 1

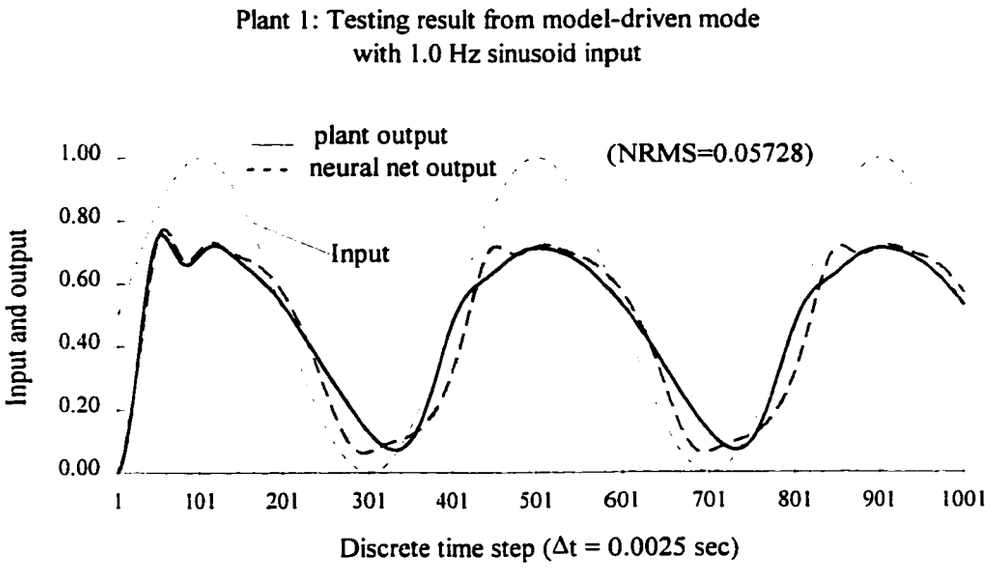


Figure 4.5(c) Model-driven testing result using sinusoidal input for Plant 1

Figure 4.5 Model-driven testing results for Plant 1

The testing results under the plant-driven mode show that the model performance is excellent in both generalizing abilities and the modeling precision. The neural model,

when subjected to new sets of data, gives excellent agreement between the net outputs and the plant outputs, as can be observed from Figures 4.4(a)-4.4(c) where the plant output curves and the model output curves can hardly be distinguished (note: the NRMS values of the tests have the same order of NRMS value of the training). The results from a large number of tests under the plant-driven mode show consistently that the obtained neural model can approximate plant outputs over the different types of fresh data with the degree of accuracy at 10^{-1} (based on the average of the errors between the plant and model outputs over the range of amplitude from 0.0 to 1.0). Using the model-driven mode, on the other hand, the testing results show that the neural net has only a good trend to follow the plant outputs, but with poorer accuracy, as can be observed from Figures 4.5(a) - 4.5(c).

Based on all of the results from both testing modes, it can be concluded that the estimated neural model has been able to capture the underlying mechanism of the dynamic plant in the range the model was trained for, i.e. the model possesses generalization capabilities. The poorer accuracy observed in the model-driven testing is caused by the recursive error accumulation. Further analysis on this aspect will be given in Chapter 5.

It should be pointed out that whether the obtained modeling accuracy is acceptable for simulation purpose, in general, depends on the accuracy requirements for the specific problems. In this thesis, it is not attempted to make the judgments on the acceptance of modeling accuracy for all of testing results since the judgment is, in fact, quite subjective.

Plant 2: The plant used in the second example is from the two papers cited [Chen, S., Billings, S. A. and Grant, P. M., 1990; Chen, S., Cowan, C. F. N., Billings, S. A. and Grant, P. M., 1990] and is expressed as:

$$y_p(k) = (0.8-0.5\exp(-y_p^2(k-1)))y_p(k-1)-(0.3+0.9\exp(-y_p^2(k-1)))y_p(k-2)+u(k-1)$$
$$+0.2u(k-2)+0.1u(k-1)u(k-2) \quad (4\text{-}5)$$

It should be noted that the sampling period for this equation was not specified. This equation itself does not indicate the sampling period and the dynamics of the physical plant the equations were trying to describe. The spectrum plot of the plant output based on pure random inputs indicated a wide bandwidth of the plant. The random numbers uniformly distributed between -0.1 and 1.1 were deemed to be appropriate inputs for the training. The reason that the data range has its extremities, -.1 and 1.1. is to ensure that the majority of the training data falls within a range of 0.0 and 1.0.

Inspection of Equation (4-5) shows that the simulated plant is highly nonlinear. A unique dynamic characteristic of this plant is stable limit cycles for a certain range of inputs. Chen and Billing have verified that the unforced response (u(t) = 0.0) of this plant with the initial conditions of $y_p(0) = 0.1$ and $y_p(-1) = 0.01$ is a stable limit cycle between
-1.0 and +1.0. It was also observed by the author of this thesis that the plant responses to a larger step input would be asymptotically stable. The critical value of the input appeared to be around 0.65. With a smaller step input, 0.5 for example, the plant response became a stable limit cycle and oscillated around the mean value of the input.

The architecture of the neural model for this plant was similar to that used for plant 1, except that the input layer consists of 5 nodes (u(k), u(k-1), u(k-2), $y_p(k-1)$ and $y_p(k-2)$). The training used 30 batches of data pairs and was terminated with the error function being reduced to about $1.15\text{x}10^{-4}$ that indicates a normalized average error of 0.005.

Training results:

The training result for the last batch of data is presented in Figure 4.6 where only 200 of 1000 data pairs are plotted for clarity and the NRMS value is evaluated over the

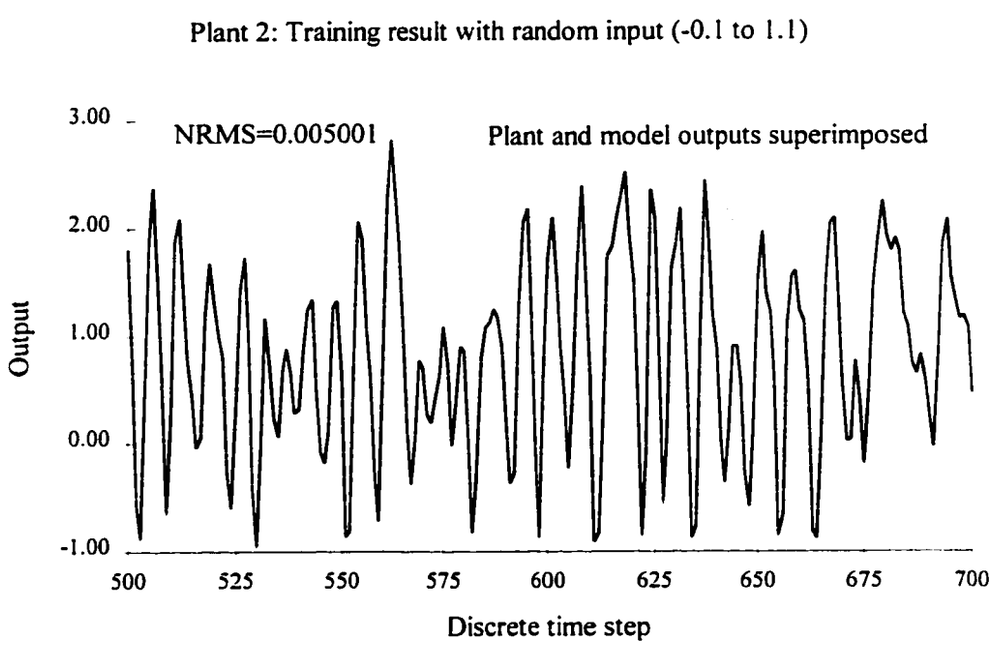


Figure 4.6 Training result for Plant 2

Testing results:

To examine the generalization capabilities and accuracy of the trained model, the neural net model was tested with various types of new data sets over the range for which the model was trained. Only three typical results are presented in Figures 4.7(a), 4.7(b), 4.7(c) for plant-driven mode, and in Figures 4.8(a), 4.8(b) and 4.8(c) for model-driven mode. The testing inputs for the results presented are random numbers between 0.0 and 1.0, step inputs of 0.5 and 1.0, and 10 Hz sinusoidal signals.

Based on an inspection of all testing results, the following observations with respect to the generalization property and modeling accuracy can be made:

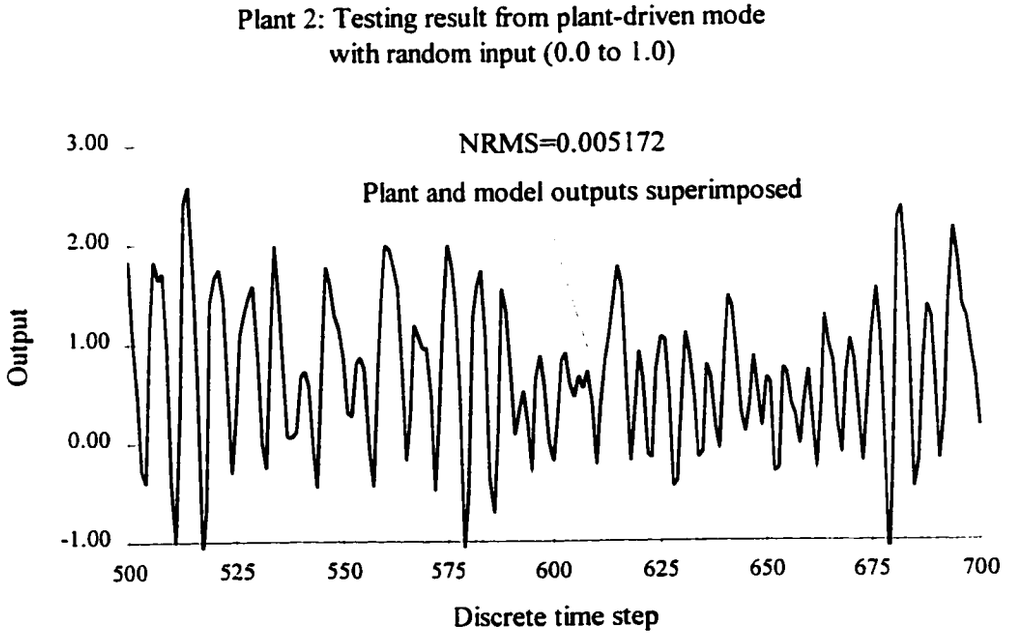


Figure 4.7(a) Plant-driven testing result using random input for Plant 2

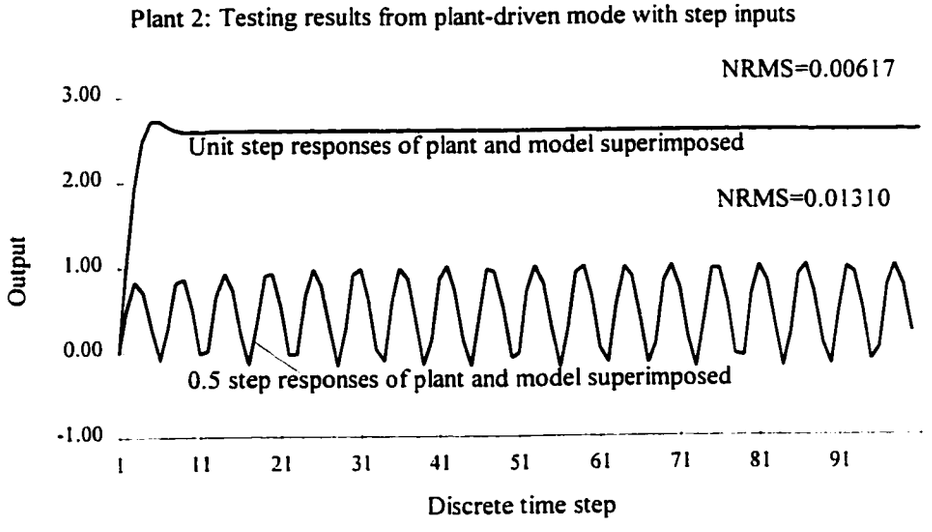


Figure 4.7(b) Plant-driven testing result using step inputs for Plant 2

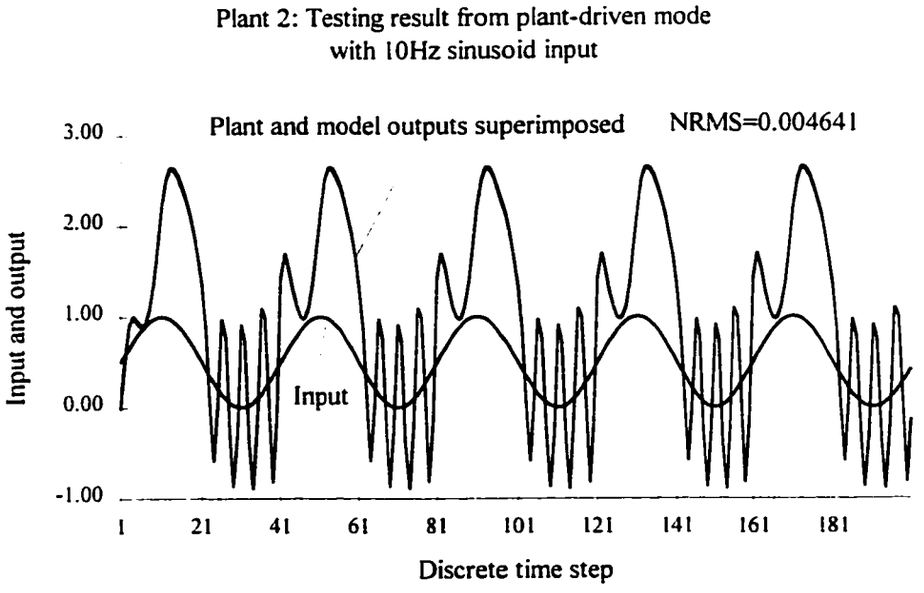


Figure 4.7(c) Plant-driven testing result using 10 Hz sinusoidal input for Plant 2

Figure 4.7 Plant-driven testing results for Plant 2

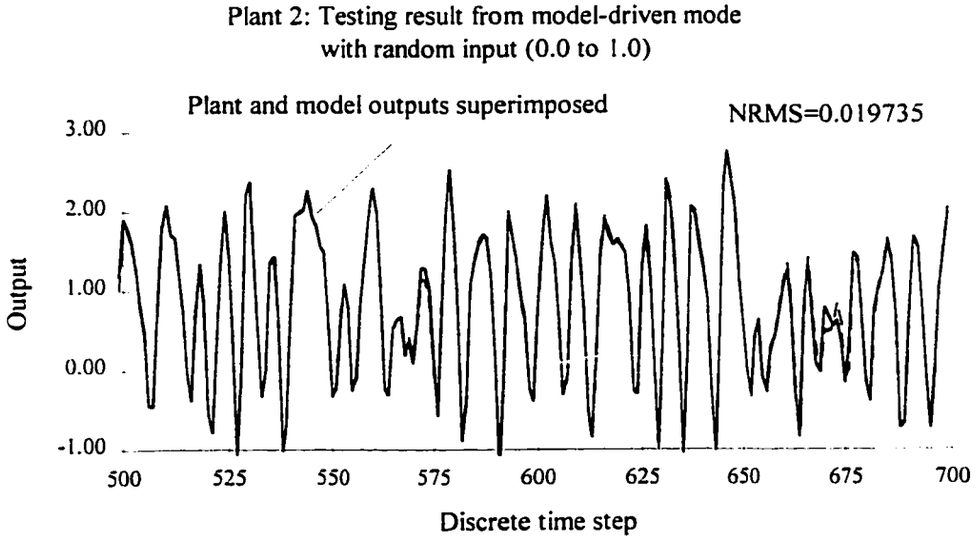


Figure 4.8(a) Model-driven testing result using random input for Plant 2

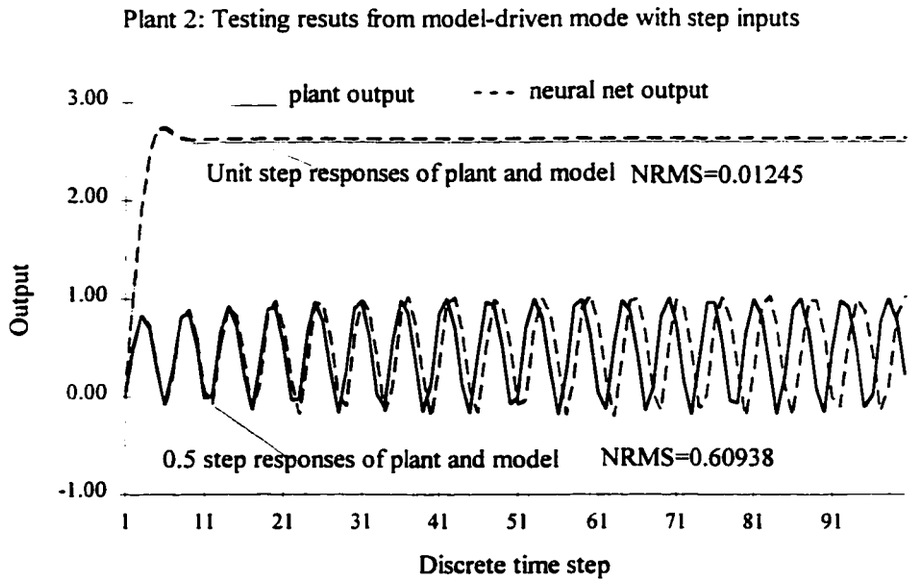


Figure 4.8(b) Model-driven testing result using step inputs for Plant 2

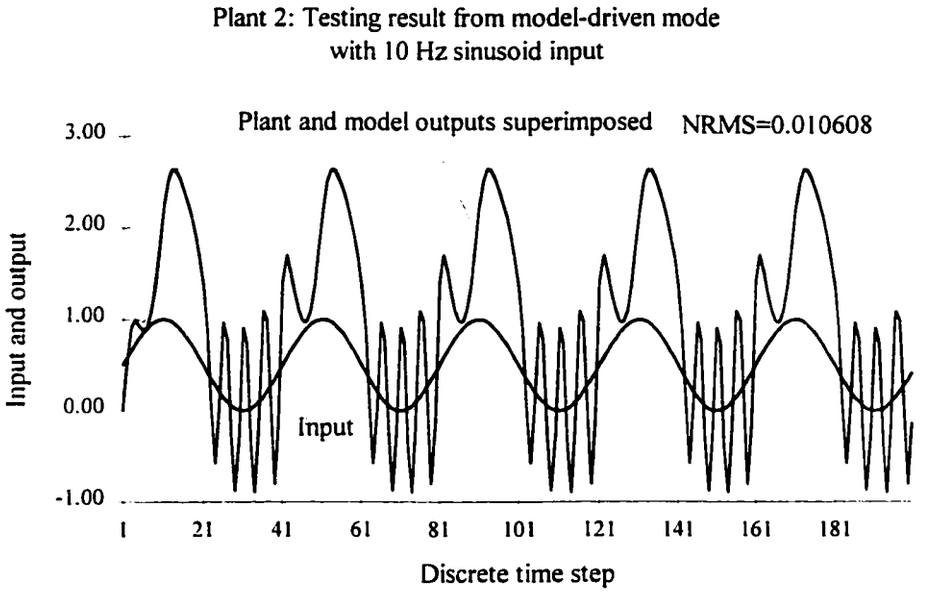


Figure 4.8(c) Model-driven testing result using 10 Hz sinusoidal input for Plant 2

Figure 4.8 Model-driven testing results for Plant 2

First of all, as the model was subjected to the various types of fresh data, it was able to predict the outputs of the plant over the whole range that the testing data covered, including the region having the complex nonlinearities, as seen in Figure 4.7(c) and Figure 4.8(c), and the region where a limit cycle was excited. This fact gives a strong evidence of the excellent generalization property that the model has gained.

Secondly, the modeling accuracy tends to be affected by testing modes and the dynamic states forced by the inputs. In the plant-driven mode, the modeling accuracy, in terms of NRMS values, remains nearly the same order as the training accuracy. Noticing that the testing data are the data that the model has not been trained for, a conclusion that the neural model is able to generalize is further supported. In the model-driven mode, the degree of the modeling accuracy appears to be dependent upon the dynamic states that the plant is driven into. Most of the testing results in this mode, with the inputs varying over the full testing range, shows a slight overall decrease in the modeling accuracy, comparing the NRMS values to the corresponding NRMS values under plant-driven mode. Whereas as the plant exhibits a stable limit cycle state, a considerable deterioration of the modeling accuracy can be observed as the plant outputs oscillate. With reference to Figure 4.8(b), for example, the NRMS value of 0.6052 from the test using 0.5 step input indicates a drastic increase in the errors, compared to the NRMS value of 0.0131 in plant-driven mode. A closer examination of Figure 4.8(b) shows that such a large modeling error is caused mainly by the phase shift between the model and plant outputs, which in turn is a consequence of the errors sequentially accumulated through the feedback of the model outputs.

It is clear then that the dynamics of this plant that is characterized by the asymptotically stable state and stable limit cycles certainly affect the modeling accuracy at least with the neural model structures used in this study.

Plant 3: In the third example, the plants that have insignificant or negligible dynamics (transients) are considered. The simplest plants can be approximated ideally by a constant-gain system equation expressed as

$$y_p(k) = u(k) \tag{4-6}$$

The purpose of using such type of a plant is to investigate how the modeling accuracy would be affected if the plant to be identified has insignificant dynamic transients. The conclusion drawn from this example may be equally applied to any nonlinear plants having negligible transients.

The structure of the neural model used is identical to the one for the second plant, i.e. there are 5 neurons at the input layer and 15 neurons at the first nonlinear hidden layer. It needs to be pointed out that in practical identification, a linear model structure would be the first choice if preliminary tests indicate that the plant is linear dominant. Use of a nonlinear model in this example is simply for illustrative purpose.

The training inputs are random numbers uniformly distributed between -0.1 and 1.1. The training shows to be a lot easier. The error function was rapidly reduced to 2.45×10^{-5} after one batch of data was fed. The training then took another 20 batches of data to refine the error function toward its minimum of 7.04×10^{-7}.

Training result:

The training result for the last batch of data is illustrated in Figure 4.9 with only 200 data pairs being plotted. The result shows that the model can reproduce the plant output over the data that the model was calculated for with excellent agreement at a small normalized RMS value of 0.001079.

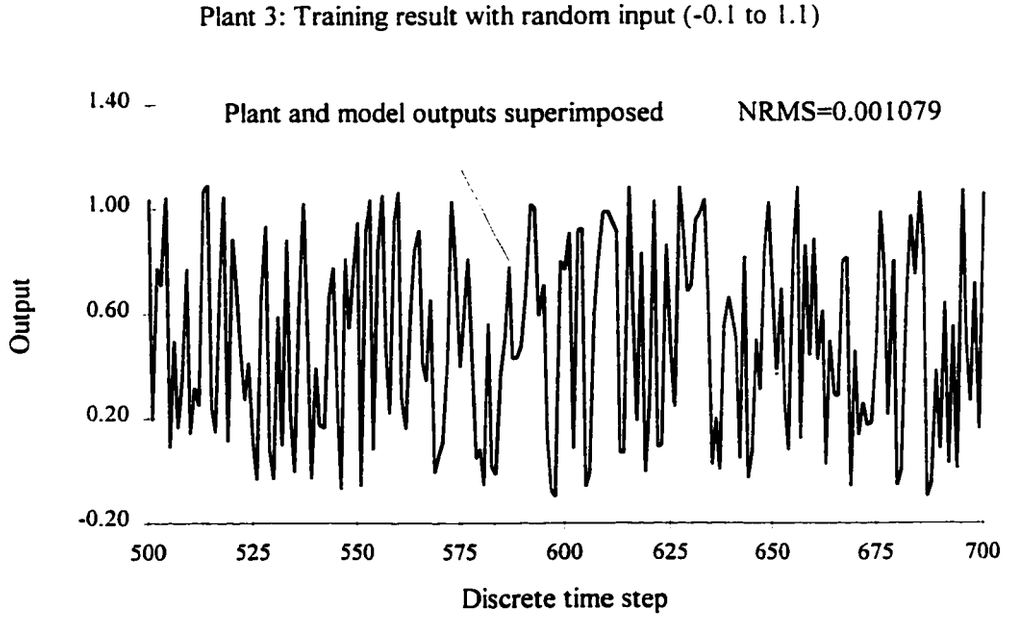


Figure 4.9 Training result for Plant 3

Testing results:

The model after estimated was extensively tested in both the plant-driven and model-driven modes using various types of inputs. Only the results using random sequences of 5 Hz bandwidth and step inputs of 0.5 and 1.0 are presented in Figures 4.10(a) and 4.10(b) for plant-driven mode and Figures 4.11(a) and 4.11(b) for model-driven mode.

All of the testing results from both modes illustrate excellent generalization capabilities of the model obtained. It is interesting to notice that compared to the previous examples where the testing accuracy significantly deteriorated as the model was tested in the model-driven mode, the accuracy observed in this example from model-driven testing remained nearly the same as from the plant-driven mode; indeed, all testing results showed consistently that testing error had the same order as the error observed in the training results. This distinct fact indicates that the neural model trained for this particular type of plant is able to simulate the plant outputs without any severe

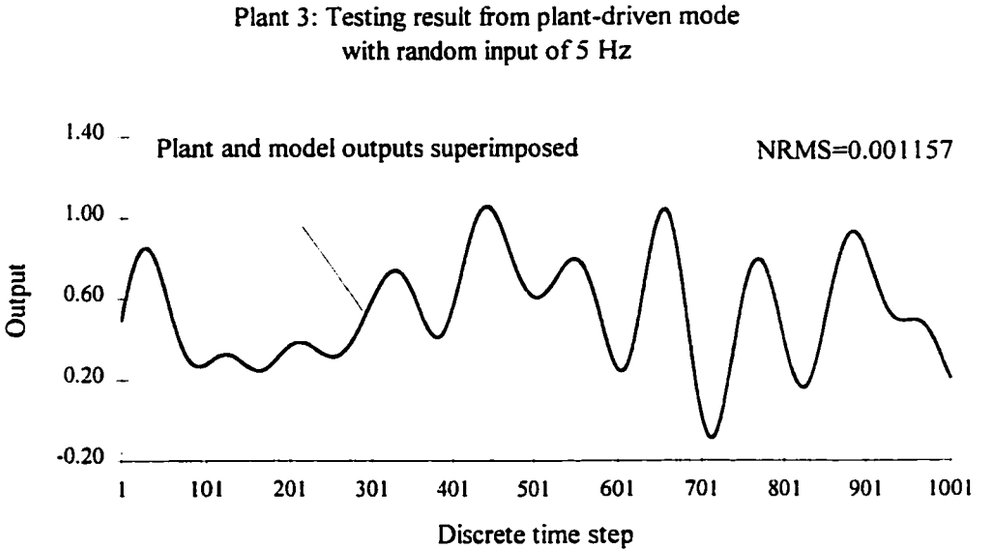


Figure 4.10(a) Plant-driven testing result using 5 Hz random numbers for Plant 3

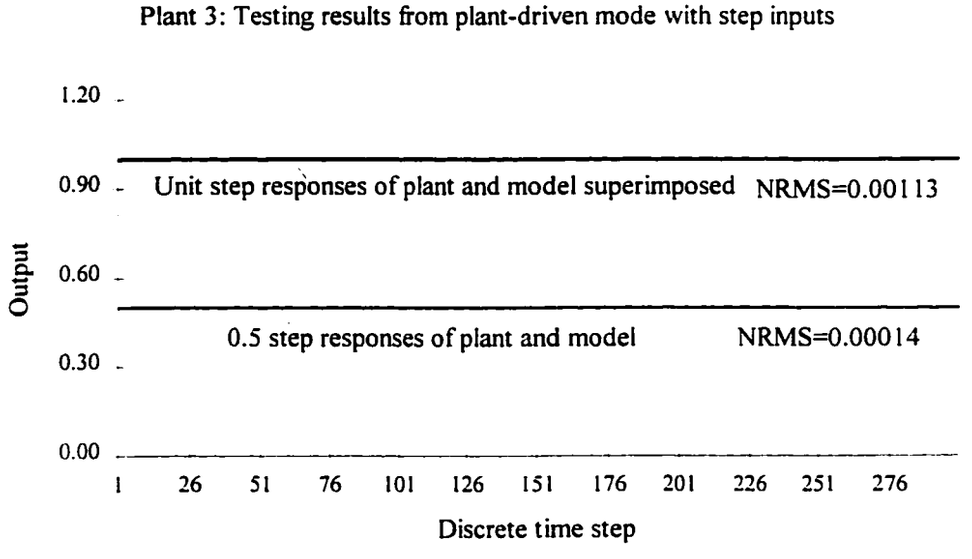


Figure 4.10(b) Plant-driven testing result using step inputs for Plant 3

Figure 4.10 Plant-driven testing results for Plant 3

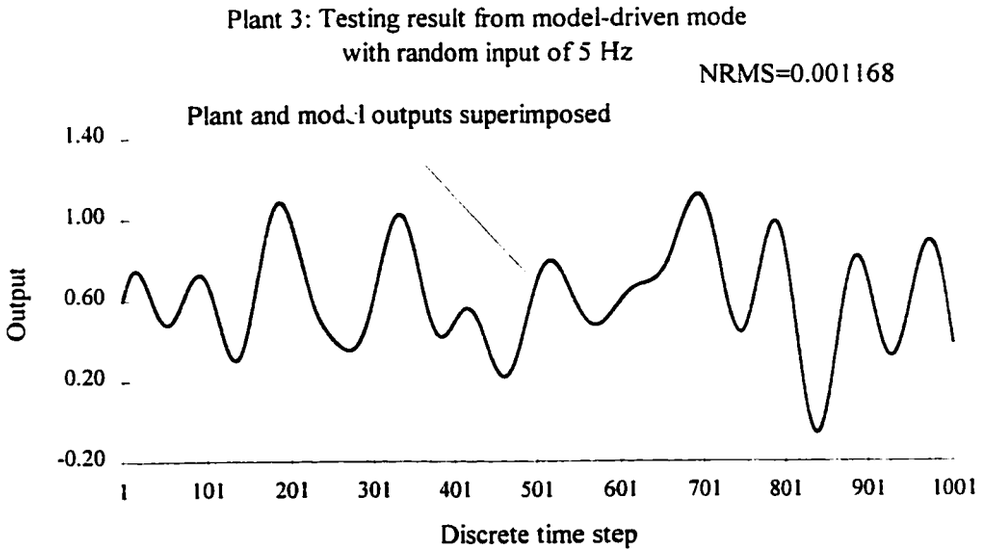


Figure 4.11(a) Model-driven testing result using 5 Hz random numbers for Plant 3

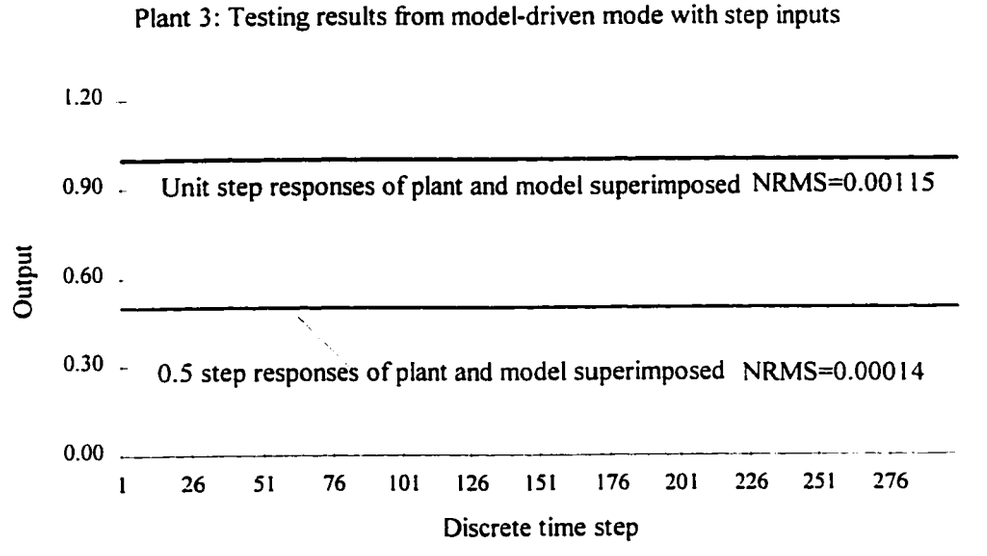


Figure 4.11(b) Model-driven testing result using step inputs for Plant 3

Figure 4.11 Model-driven testing results for Plant 3

degradation of the modeling accuracy even though the model structure does include feedback paths.

4.6 Discussion

It has been found through extensive simulation studies that in nearly all cases, the structure errors in plant-driven testing, $e_{sp}(k)$, are almost the same in magnitude as ones found from their corresponding training results where the best fitting of the models to noise-free data has been ascertained. This is conceivable because the models in both training and plant-driven testing use delayed plant outputs. In model-driven testing, the structure errors, $e_{sm}(k)$, exhibit, in some cases, significant growth as was observed, for example, in the first simulation example. This is because as the past model outputs are used to predict the current model output, the errors associated with these past outputs are also brought back simultaneously into the model, accumulated and embedded into the subsequent model outputs. Such an error accumulation through the model feedback paths will, in general, deteriorate the overall modeling accuracy in the simulation mode. The degree of this deterioration depends obviously on the final training accuracy, and the apparent plant dynamics for which the model is estimated.

Chapter 5

Model Structure Error Analysis

In this Chapter, the model structure error observed in simulation examples presented in Chapter 4 will be further analyzed. The purpose of this analysis is to examine how the magnitude of the accumulated modeling errors $e_{sm}(k)$ are influenced by final training accuracy and plant dynamics. Error analysis will be attempted by establishing the relationship between $e_{sm}(k)$ and $e_{sp}(k)$ (a reflection of training errors) via parameters related to the dynamics of the plants (in discrete form). In the last part of this Chapter, the results will be used to interpret the modeling errors, $e_{sm}(k)$, shown differently in the three simulation examples.

5.1 Analysis

A thorough theoretical analysis of modeling errors is complex because of the presence of nonlinearities in neural networks and of the numerical iterative estimation procedure used for the training. To simplify the analysis, a linear model and a linear plant are considered. The conclusions and the comments drawn from these results can be applied to most nonlinear cases since many nonlinear functions can be, at least in principle, expanded by Taylor's series about the reference points and can be expressed approximately in linear form (except that the coefficients in the linearized functions vary with reference points).

Assume a linear plant that can be realized by a difference equation of the form

$$y_p(k) = \sum_{i=1}^{n} a_i y_p(k-i) + \sum_{j=0}^{m} b_j u(k-j) \qquad (5\text{-}1)$$

where n and m are integers, $n \geq 1$ and $m \geq 0$.

Consider a simple case where n = 2 and m = 0. Equation (5-1) then becomes a realization of a second order plant in discrete form:

$$y_p(k) = a_1 y_p(k-1) + a_2 y_p(k-2) + b_0 u(k) \tag{5-2}$$

For an asymptotically stable linear plant, the coefficients, a_1 and a_2 are such that the roots of the plant characteristic equation are located inside the unit circle in the Z-plane. Note that for a dynamic plant expressed in discrete form, the coefficients, such as a_1 and a_2 in Equation (5-2), are essentially associated with both the plant dynamic characteristics before discretization and the sampling rate. The dynamics described by the difference equation coefficients include, therefore, the effects of the dynamics observed in the continuous domain and sampling frequencies. This is a critical aspect in analyzing the effects of these coefficients on modeling accuracy as will be shown later.

The linear neural net model can be readily obtained by simplifying the sigmoidal function at each neuron to a unit gain. The estimated linear model for the plant considered can be expressed by

$$y_{m,m}(k) = \bar{a}_1 y_m(k-1) + \bar{a}_2 y_m(k-2) + \bar{b}_0 u(k) \tag{5-3}$$

for the model used in model-driven mode (simulation mode), and by

$$y_{m,p}(k) = \bar{a}_1 y_p(k-1) + \bar{a}_2 y_p(k-2) + \bar{b}_0 u(k) \tag{5-4}$$

for the model in plant-driven mode, where, $\bar{a}_1$, $\bar{a}_2$ and $\bar{b}_0$ are estimates of the coefficients a_1, a_2 and b_0 in Equation (5-2), and are often referred to as the model parameters.

The simulation accuracy is evaluated by estimating the modeling error in the model-driven mode, $e_{sm}(k)$. A straightforward mathematical derivation gives an analytical expression of $e_{sm}(k)$ as a function of $e_{sp}(k)$ where $e_{sp}(k)$ represents the degree of the training accuracy.

$$e_{sm}(k) = e_{sp}(k), \qquad k=1 \qquad (5\text{-}5\text{-}1)$$

$$e_{sm}(k) = e_{sp}(k) + \bar{a}_1 e_{sm}(k-1), \qquad k=2 \qquad (5\text{-}5\text{-}2)$$

$$e_{sm}(k) = e_{sp}(k) + \bar{a}_1 e_{sm}(k-1) + \bar{a}_2 e_{sm}(k-2), \quad k\geq 3 \qquad (5\text{-}5\text{-}3)$$

where $e_{sm}(k)$ and $e_{sp}(k)$ are given by

$$e_{sm}(k) = y_p(k) - y_{m,m}(k) \qquad (5\text{-}6)$$

and

$$e_{sp}(k) = y_p(k) - y_{m,p}(k) \qquad (5\text{-}7)$$

respectively.

Inspection of Equation (5-5) shows that at each time instant k, the modeling error $e_{sm}(k)$ consists of two parts, a current $e_{sp}(k)$, and its previously accumulated errors combined through the feedback coefficients $\bar{a}_1$, $\bar{a}_2$. The resultant modeling error $e_{sm}(k)$ will be, in general, larger in magnitude than that of $e_{sp}(k)$, unless $\bar{a}_1 e_{sm}(k-1)$ and $\bar{a}_2 e_{sm}(k-2)$ in Equation (5-5-3) happen to cancel each other out, which is very unlikely in practice. Since the coefficients $\bar{a}_1$ and $\bar{a}_2$ are the approximation of the plant coefficients a_1 and a_2, the dynamics of the given plant certainly affect the error accumulation process. This proves fundamentally that the modeling accuracy for the simulation model (represented by $e_{sm}(k)$ in Equation (5-5)) is determined by both training accuracy (represented by $e_{sp}(k)$) and the plant dynamics (reflected in difference equation coefficients) that the model tends to simulate.

5.2 Effects of Training Accuracy on Modeling Errors

This fundamental result may suggest that a further improvement on the accuracy in model simulation could be achieved through further refining the training accuracy (equivalent to further reducing the modeling error $e_{sp}(k)$ in Equation (5-5)) during the parameter estimation stage, since a smaller error $e_{sp}(k)$ will give a smaller accumulated error $e_{sm}(k)$ as indicated by Equation (5-5). This suggestion is also based on a proven theoretical result that the class of neural models is able to approximate any continuous

functions to any degree of accuracy, provided enough hidden units are available. This aspect has been investigated experimentally by increasing the number of neurons in the hidden layer by the author. The results, however, show no obvious improvement on the training accuracy. This has raised an interesting issue on the numerically achievable accuracy of the neural model class as well as other types of model classes that have been described mathematically to have the property of approximating a system to within an arbitrary accuracy. This part of the study has demonstrated that in practical implementations, there is always a ultimate limitation on the training accuracy for a selected model class and the plant under identification, due to the limits on the finite order of variables in the models, complexity of nonlinear plants, and inefficiency in nonlinear estimation algorithms, etc. (Preliminary studies indicated that the errors caused by finite precision and round-off errors in numerical computing are very small (less than 10^{-14}) compared to modeling errors (10^{-8}), and were not considered as a major part of error source.)

The simulation work has shown that the conjugate gradient training algorithm always gives monotonically and rapidly converging solutions towards the minimum of the error functions, and further refinement of training accuracy is limited mainly by how well the model structure matches the plant structure represented by input-output data information. The neural model class is just one special type of model structure. Although it has been proved to be universal in approximating wide varieties of nonlinear continuous functions with a finite number of neurons, the final training accuracy will be always confined to a certain degree. This means that there is a practical limitation on its inherent approximation capabilities for any given plant (except the case where the plant structure happens to be identical to that of the neural model). This also implies that if, for a given plant and a neural model, the modeling error $e_{sm}(k)$ appears to be unacceptable, then any further effective improvement on the modeling accuracy through further reducing training errors without modifying or changing the model structure would be very difficult, if not impossible, because the accuracy has been ultimately

limited by the model structure itself. A better alternative is to re-select the model structure that has a closer match to the relationship presented in input-output data.

It is worthwhile to notice that the training accuracy (reflected by $e_{sp}(k)$ in Equation (5-5)) observed from the identification of several different nonlinear dynamic plants using the structure of partially recurrent neural nets has been able to reach a degree of less than 5% in terms of NRMS values, which is often considered to be satisfactory or accurate enough in engineering practice. As a matter of fact, the deterioration of the modeling error $e_{sm}(k)$ in the simulation mode (model-driven) occurs only as the combined effects of a_1 and a_2 in Equation (5-5) on the recursive accumulation of the errors become significant.

5.3 Effects of Apparent Plant Dynamic Characteristics

In an attempt to further examine the effects of plant apparent dynamics reflected by the difference equation coefficients on the growth of the modeling error $e_{sm}(k)$, the Z-transform of Equation (5-5-3) is taken, yielding the error transfer function G(z):

$$G(z) = \frac{e_{sm}(z)}{e_{sp}(z)} = \frac{z^2}{z^2 - \bar{a}_1 z - \bar{a}_2} \tag{5-8}$$

Notice that as the coefficients $\bar{a}_1$, $\bar{a}_2$ are approximates of a_1 and a_2, the denominator of the error transfer function can be considered as approximately the same as that of the plant transfer function. In fact, the differences in $e_{sm}(k)$ caused by using plant parameters a_1, a_2 and by using the well estimated model parameters $\bar{a}_1$, $\bar{a}_2$ are very small compared to the absolute magnitude of $e_{sm}(k)$, and can be neglected. The error transfer function can then be written using plant parameters a_1 and a_2 for simplicity in analysis.

$$G(z) = \frac{e_{sm}(z)}{e_{sp}(z)} = \frac{z^2}{z^2 - a_1 z - a_2} \tag{5-9}$$

Let A(z) and B(z) denote the denominator and numerator of the transfer function, respectively. From discrete system theory, it is known that for an error transfer function

having no repeated poles, the response of $e_{sm}(k)$ to a unit step input of $e_{sp}(k)$ (an unrealistic form of input in reality) can be estimated by:

$$e_{sm}(k) = \frac{B(1)}{A(1)} + \frac{B(p_{r1})}{(p_{r1} - 1.0)\dot{A}(p_{r1})} p_{r1}{}^{k} + \frac{B(p_{r2})}{(p_{r2} - 1.0)\dot{A}(p_{r2})} p_{r2}{}^{k} \qquad (5\text{-}10\text{-}1)$$

if the function has two real roots, or

$$e_{sm}(k) = \frac{B(1)}{A(1)} + 2\left|\frac{B(p_c)}{(p_c - 1.0)\dot{A}(p_c)}\right| \cdot |p_c|^{k} \cos(k\theta_c + \phi_c) \qquad (5\text{-}10\text{-}2)$$

if the function has a pair of conjugate complex roots.

In these equations,

p_r: real roots;

p_c: complex conjugate roots with α and β denoting their real and imaginary parts, respectively.

$$\dot{A}(p_r) = \left.\frac{dA(z)}{dz}\right|_{z=p_r};$$

$$\theta_c = \tan^{-1}\frac{\beta}{\alpha};$$

$$\phi_c = \angle B(p_c) - \angle(p_c - 1.0) - \angle\dot{A}(p_c).$$

The first term on the right hand side of Equations (5-10-1) and (5-10-2) represents the steady state value of the response, and the remaining term(s) are the transient parts of the responses. Rewriting the steady state part in forms of the two different roots p_1 and p_2 yields:

$$C_{s.s.} = \frac{B(1)}{A(1)} = \frac{1}{(1.0 - p_1)(1.0 - p_2)} \qquad (5\text{-}11)$$

Equation (5-11) shows clearly that the worst scenario in which $e_{sm}(k)$ gains the largest value occurs when either of the roots is nearing or at the point (1.0, 0.0j) in the Z-plane.

Consider the transient part in Equations (5-10-1) and (5-10-2). After a very few steps of mathematical manipulation, it can be shown that the worst scenario occurs at the

location (1.0, 0j) of the Z-plane. As one of the real roots or the complex roots approaches that location, the denominator of the transient term(s) approaches to zero, resulting in extremely large transient responses (approaching infinite). In addition, the decay of the transient will be slow because the exponential terms in Equations (5-10-1) and (5-10-2) will decrease slowly with integer time step *k*.

The accumulation of modeling error $e_{sm}(k)$ is attributed to the roots of the plant characteristic equation that are nearing the location (1.0, 0.0j) in the Z-plane. Notice the fact that the roots of the Z-transfer function are affected by the inherent dynamics of the plant and the sampling time interval. It appears that an effective way to reduce the modeling error $e_{sm}(k)$ is to move the roots away from the point (1.0, 0.0j) by changing (reducing) sampling speed, provided that the frequency content will still be sufficient.

To illustrate this concept, consider a plant that can be realized by a standard linear 2nd order underdamped system with its natural frequency at about 5 Hz and damping ratio of 0.316 (specified in continuous time domain). A linear neural model was trained using the plant's input-output data sampled at a relatively high speed of 400 Hz. For the purpose of simulation, the plant outputs corresponding to 400 Hz sampling speed were approximated by a difference equation (converted from the differential equation using backward differentiation scheme):

$$y_P(k) = 1.94582y_P(k-1) - 0.95177y_P(k-2) + 0.00595u(k) \qquad (5\text{-}12)$$

The training process was terminated when the training error was reduced to the order of 10^{-4} on the average. The obtained model was expressed by the difference equation with estimated parameters:

$$y_n(k) = 1.94099y_n(k-1) - 0.94705y_n(k-2) + 0.00606u(k) \qquad (5\text{-}13)$$

The model was then subjected to several different types of inputs to test its accuracy in both plant driven and model-driven modes. Table 5.1 summarizes two typical testing results from inputs of sinusoid and unit step types, and compares the

modeling errors in terms of NRMS values over 1000 data points. The error accumulation is significant because of the large ratio of NRMS values in the simulation mode to that in plant-driven mode.

Examination of root locations associated with the difference Equation (5-12) or (5-13) reveals that the plant (in the discrete time domain) or its estimated model has a pair of complex conjugate roots at 0.97050 ± 0.07204j (calculated from the model parameters), very close to the critical point (1.0, 0.0j).

Table 5.1 Linear plant modeling: testing results with sampling speed of 400 Hz

Inputs	NRMS (plant-driven)	NRMS (model-driven)	Error ratio
5 Hz sinusoid	0.00011	0.02483	225.73
Unit step	0.00003	0.00581	193.67

In an attempt to move the roots farther away from the point (1.0, 0.0j), hence reducing the simulation error $e_{sm}(k)$, a lower sampling speed of 75 Hz (still sufficiently high considering the plant's natural frequency of 5 Hz) was then used. For simplicity in simulation, instead of sampling the plant outputs generated from the original equation, a modified difference equation given by Equation (5-14) was used to approximate the sampled data, which gave similar dynamic characteristics to those in Equation (5-12) except the time interval associated with the sampling speed of 75 Hz.

$$y_p(k) = 1.655y_p(k-1) - 0.782y_p(k-2) + 0.127u(k) \qquad (5\text{-}14)$$

The training of its linear neural model was terminated when the training errors decreased to 3.04 $x10^{-4}$ on average over 1000 points. The mathematical expression of the model was obtained as:

$$y(k) = 1.65054y(k-1) - 0.77808y(k-2) + 0.12774u(k) \qquad (5\text{-}15)$$

Notice that the roots have been relocated to the position of $(0.82527 \pm 0.31146j)$.

Again, the model was tested to observe the modeling accuracy in both modes. The results from using sinusoid and unit step inputs are presented in Table 5.2 in a similar manner as in Table 5.1[1]. A comparison of the error ratio in Table 5.2 to that in Table 5.1 clearly shows that the error accumulation has been considerably reduced from over 100 times to less than 11 times by a slower sampling. This is essentially equivalent to adjusting the steady state gain of the linear error transfer function down to a much smaller value. It needs to be mentioned that for the linear plant considered here, the problem of having an extremely large gain in the error transfer function (due to very fast sampling speed) resulting in excessive error responses did not appear in the plant transfer function even though both transfer functions have the same denominator. The reason is that in the plant transfer function, high sampling speeds also simultaneously affect the gain in the numerator of the transfer function, giving rise to a much smaller gain. Overall, the effect of high sampling speeds on the response of the plant would then be nearly canceled out.

Table 5.2 Linear plant modeling: testing results with sampling speed of 75 Hz

Inputs	NRMS (plant-driven)	NRMS (model-driven)	Error ratio
5 Hz sinusoid	0.00062	0.00650	10.56
Unit step	0.00016	0.00122	7.50

The above fundamental results are derived based on the linear model and linear plant, but the principle revealed by this simplified analysis can be applied to interpret, qualitatively, the modeling errors observed in identifications using nonlinear neural models.

[1] Slightly larger NRMS values in plant-driven mode in Table 5.2 than those in Table 5.1 are due to the fact that the training with data sampled at 75 Hz speed was forced to stop slightly earlier than the training with 400 Hz sampling speed. Since in this illustrative example, the emphasis is on the relationship of $e_{sm}(k)$ and $e_{sp}(k)$, represented as the ratio NRMS values in Tables 5.1 and 5.2, slightly earlier termination of the training would not, in principle, affect the analysis results, as the level of the training errors only represents reference bases of $e_{sp}(k)$ to which the accumulated $e_{sm}(k)$ is compared.

5.4 Interpretation of Simulation Results

Revisit the first simulation example where a nonlinear dynamic plant was modeled using a nonlinear neural network, and the sampling speed needed to be high enough to accurately describe the transient part of the plant dynamic response. The identification result with 400 Hz sampling speed (almost 80 times the plant natural frequency) showed that the magnitude of the modeling error in the simulation mode was over 100 times of those in the plant driven mode in terms of the ratio of NRMS values, and the modeling accuracy so obtained was considered to be insufficient. Based on the above analysis result, the suspected cause for the large accumulated errors $e_{sm}(k)$ was an unnecessarily high sampling speed. To investigate this aspect, a lower sampling speed of 75 Hz (about 15 times the plant natural frequency) was then adopted. The plant outputs corresponding to this sampling frequency were approximated by a modified difference equation:

$$y_p(k) = \frac{2.1133y_p(k-1) + 0.1244y_p^2(k-1) - y_p(k-2) + 0.0622u(k)}{1.1133 + 0.2489|y_p(k-1)|} \qquad (5\text{-}16)$$

The training of the neural model with the same structure was terminated when the average of training errors decreased to the order of $1.15\text{x}10^{-3}$. Testing results with various types of inputs shows that the ratio of NRMS values in the simulation mode to that in the plant-driven mode drastically decreased from over 100 to about 5 to 15. The testing results using random numbers with 5 Hz bandwidth, step inputs (1.0 , 0.5, 0.25) and 1 Hz sinusoid are summarized in Table 5.3 and plotted in Figures 5.1(a)-(c). For comparison, the error ratios from the model trained using the sampling speed of 400 Hz were also tabulated in Table 5.4. Regardless of the observed fact that training with 75 Hz data sampling speed seemed to be slightly harder and the training accuracy was somewhat poorer, the overall modeling accuracy in the simulation mode was significantly improved as can be observed through a comparison of Table 5.3 to Table 5.4, and Figures 5.1(a)-(c) to Figures 4.5(a)-(c).

Table 5.3 Nonlinear plant modeling: testing results with sampling speed of 75 Hz

Inputs	NRMS (plant-driven)	NRMS (model-driven)	Error ratio
5 Hz random	0.00098	0.01127	11.50
1.0	0.00191	0.01255	6.57
0.5	0.00060	0.00566	9.43
0.25	0.00174	0.02653	15.23
1 Hz sinusoid	0.00106	0.01236	11.66

Table 5.4 Nonlinear plant modeling: testing results with sampling speed of 400 Hz

Inputs	NRMS (plant-driven)	NRMS (model-driven)	Error ratio
5 Hz random	0.000278	0.06319	227.30
1.0	0.00015	0.01502	100.13
0.5	0.00018	0.03684	204.67
0.25	0.00080	0.30816	385.20
1 Hz sinusoid	0.00022	0.05728	260.36

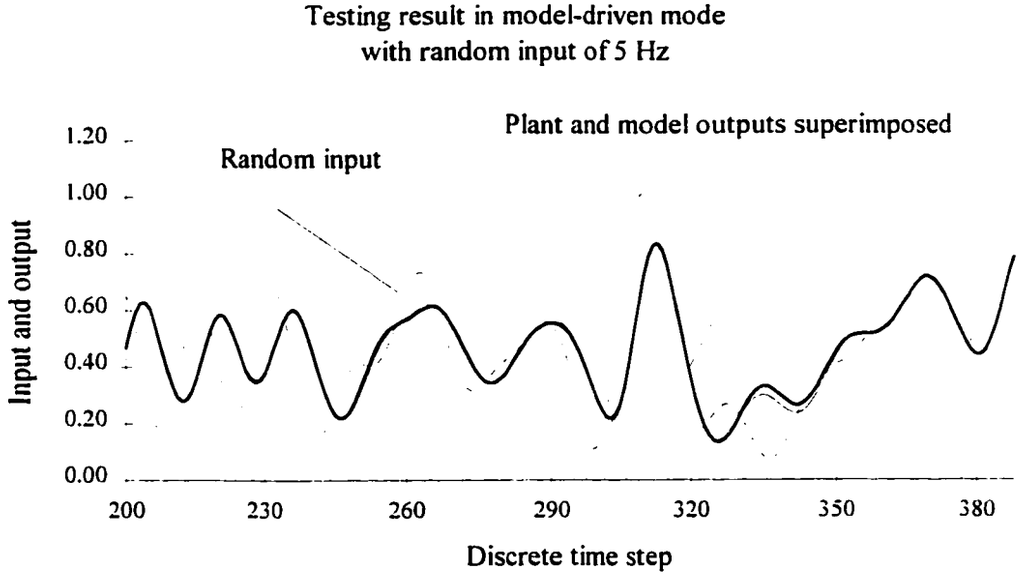


Figure 5.1 (a) Testing result with 5 Hz random input

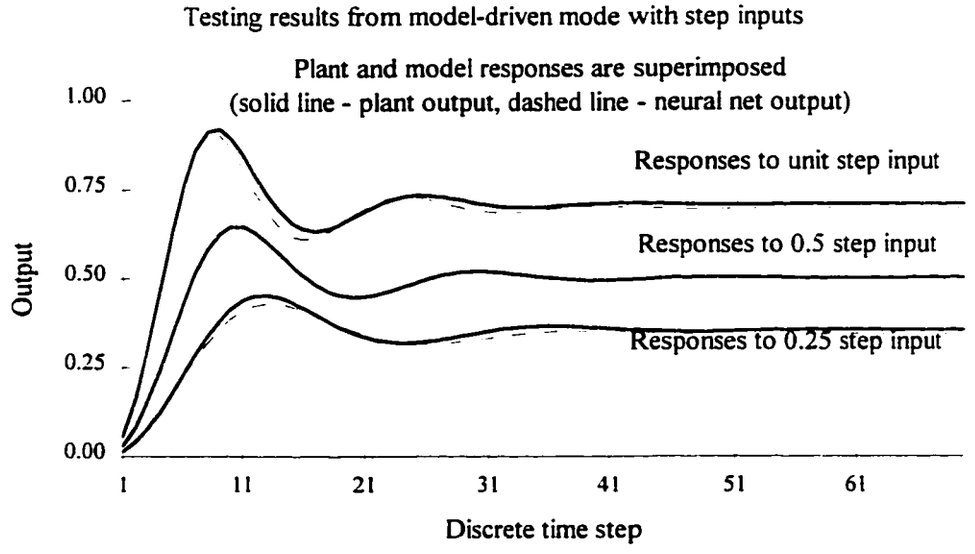


Figure 5.1(b) Testing result with 3 step inputs

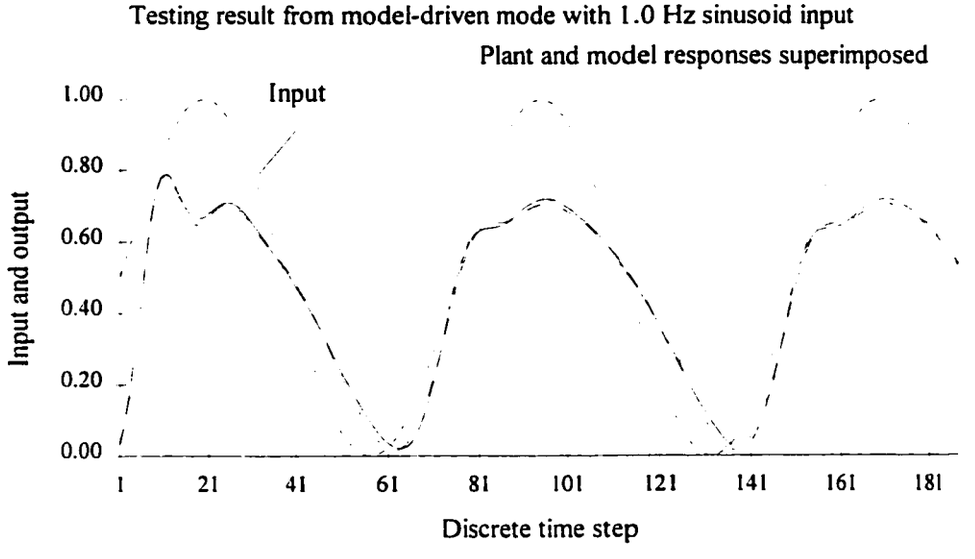


Figure 5.1(c) Testing result with 1 Hz sinusoid input

Figure 5.1 Model-driven testing results for nonlinear plant with sampling speed of 75 Hz

Consider now the second simulation example, where the plant that was modeled by a neural model, was highly nonlinear with two distinct equilibrium states: an asymptotically stable state excited with a larger initial input, and a limit cycle state with small initial conditions (small input or unforced system response). The identification results showed that in the stable state, the modeling accuracy was reasonably good. For instance, with a unit step input in a model testing, the NRMS value in plant-driven mode was 0.00617, and 0.01245 in model-driven model, resulting in an error ratio of 2.02. In the limit cycle state, the modeling accuracy in the model-driven mode was considerably degraded from a NRMS value of 0.01310 in the plant-driven mode to 0.6052 in the model-driven mode, giving a ratio of 46.20, as observed in Figure 4.7(b) and Figure 4.8(b).

To further examine the properties of modeling errors, consider first the plant dynamics under larger initial conditions. Careful inspection of the plant expression shows that this nonlinear expression can be simplified to a linear form given by

$$y_p(k) = 0.8y_p(k-1) - 0.3y_p(k-2) + x(k) \tag{5-17}$$

where $x(k) = u(k-1) + 0.2u(k-2) + 0.1u(k-1)u(k-2)$

With a_1=0.8 and a_2=-0.3, the corresponding error equation is given by

$$e_{sm}(k) = e_{sp}(k) + 0.8e_{sm}(k-1) - 0.3e_{sm}(k-2) \tag{5-18}$$

The roots of its discrete characteristic equation can be readily calculated at 0.4± 0.374j in the Z-plane, which is certainly away from the critical point: 1.0, 0j. As a consequence, the gain of the error function (associated with error ratio) will not be excessive. This can be illustrated by the estimate of the ratio of the steady state response of error $e_{sm}(k)$ to a standard unit step input of $e_{sp}(k)$ using the final-value theorem of the Z-transform. The estimated gain is 2, approximating to the error ratio, 2.02, observed in the testing result using a unit step input.

Under small initial conditions, the plant is essentially in an unstable state. A simplified analysis through linearizing the plant expression indicates that the

corresponding error function would have a pair of complex conjugate roots outside of the unit circle, nearly on the imaginary axis, in the Z-plane. The resulting error response $e_{sm}(k)$ to a small step input of $e_{sp}(k)$, for example, would be an exponentially diverging sinusoid until it reaches the equilibrium state of a limit cycle. The accumulated error $e_{sm}(k)$ cannot be exactly estimated due to the nonlinearities and invalidity of using the final-value theorem, but it can be intuitively explained as follows: The major cause for the error growth in the simulation mode is due to positive damping (not the root location nearing the critical point, 1.0, 0.0j). In the unstable state, the error accumulates exponentially with discrete time step increases. Its magnitude will be, however, constrained due to the existence of the limit cycle, resulting in only a significant phase shift being observed.

Examine the third example that simulates a special and simple case where the plants have their characteristic equation roots quite close to or nearly at the origin of the Z-plane (both a_1 and a_2 in Equation (5-2) being very small approximating to zero). From discrete system theory, it is known that as the roots approach the origin, the transient part of a plant response will decay rapidly, and when the roots reach the origin, the transient part will completely disappear, leaving out only the steady state of the response. This case was simulated by an ideally simplified plant expression $y(k) = u(k)$, indicating $a_1=0$ and $a_2=0$. The estimated modeling error $e_{sm}(k)$ based on Equation (5-5) would be $e_{sm}(k) = e_{sp}(k)$, implying that the modeling accuracy in simulation remains at the same degree as training accuracy. This is exactly what has been observed from the testing results where the modeling errors, as represented by NRMS values in Figures 4.10 and 4.11, have nearly the same order in magnitude in the plant-driven mode (NRMS = 0.001157 with random input of 5 Hz bandwidth) as in the model-driven mode (NRMS = 0.001168 with random input of 5 Hz bandwidth).

The deterioration of the modeling accuracy in the model-driven mode was not observed although the network model does have external feedback paths. This case approximately simulates a class of practical situations where the time period for the

transient part of the plant response is considered to be insignificant compared to the time scale used, and can be ignored. The sampling speeds under these situations are often relatively slow (not enough to capture the transients of dynamics as it may not be necessary) resulting in the root locations nearing the origin in Z-plane. The plant dynamics in discrete time domain will then appear to have transients decay completely in one or very few limited discrete time steps. It can be concluded from this simplified case that for plants having very insignificant or negligible transient parts in their dynamic responses, the modeling accuracy will not be degraded when the model is used in simulation mode.

5.5 Comments

The error analysis presented in this chapter has provided an insight into the mechanism on how the modeling error of a recurrent type of model is affected by the training accuracy associated with the structure differences between the plant and the model, and by the plant apparent dynamics (transient state). High training accuracy is desirable (for accurate modeling results), but always, in practice, limited to a certain degree, because of the limited neural net models' inherent approximation capabilities. The analysis reveals that the feedback paths in the recurrent model structures introduce the dynamics of the model, but, at the same time may cause the error accumulation problem. The degree of the error accumulation depends on the plant transient property and the sampling rate. Given a dynamic plant, the sampling time period needs to be chosen properly with respect to the "nominal time constant" of the plant. The sampling rate needs to be high enough to obtain sufficient discrete points in the transient state of the plant response, but unnecessarily high rates should be avoided in order to reduce the error accumulation.

Chapter 6
Experimental System

6.1 Introduction

In the previous Chapters, the concept of using a particular class of neural network models to identify a system based on the approximation approach has been presented. The feasibility of using a partially recurrent neural network to model some nonlinear system dynamics was examined through simulation examples and modeling error analysis. Particular attention has been given to the modeling error behavior under noise-free conditions, at which the modeling error was solely caused by the differences between the model structure and system structure (though unknown in reality), in order to examine the ultimately achievable modeling accuracy for a given system. It has been ascertained that for an adequately chosen network and a stable plant, if care is taken in the sample rate used in the modeling process, the established model can provide a reasonable representation of the plant over a sufficient frequency band and amplitude range.

In this and the next chapters, it is intended to demonstrate that the concept and approach can be successfully applied to the modeling of a physical dynamic system. From an experimental point view, this part of the study examines the practical quality of the model for the real system, which is affected by the compounding of modeling error and unpredictable data noise, and the physical constraints imposed by the experimental facilities available for this study.

The choice of the plant to be modeled was dictated by several factors: availability, reasonable bandwidth, and nonlinear behavior. In addition, it was desirable to use a component which was very complex to model using traditional analytical methods (thus enforcing the benefits of the proposed modeling method). It was decided to choose a load sensing pump[1] after examining several hydraulic components (variable displacement pump, relief valves, servo valves, actuators etc.).

This Chapter, then, describes the operation of the load sensing pump, the pump model structure that was chosen to facilitate the test, the configuration of the experimental system, data acquisition system, transducer calibrations and testing procedure.

6.2 Load Sensing System

A load sensing pump is designed to maintain a fixed pressure drop across a controlling orifice in order to control flow (pressure compensated flow control system) in an energy efficient manner [Mollo, J. R., 1990]. A schematic circuit of the pump and resistive load is illustrated in Figure 6.1. With reference to this figure, P_s is defined as the pump output pressure, P_l as the load pressure, P_c as the control pressure, Q_p as the pump flow, ω as the pump shaft speed and τ as the shaft torque. In the following description, assume the system is at steady state. The compensator (1 in Figure 6.1) is at the closed position shown. A force balance between hydraulic pressure (P_s and P_l) and the spring force exists. Let P_l increase from the steady state condition. First, the pressure drop (P_s - P_l) across the load orifice (6 in Figure 6.1) decreases which results in a decrease in Q_l. Meanwhile, the increase in P_l is sensed at the compensator and creates a force imbalance across the compensator spool. This causes the spool to be pushed to the right. The spool orifice (3 in Figure 6.1) opens and ports pressurized fluid to tank.

[1] The pump used in this study was a John Deere AL75305 pump. This pump and test stand were built from funds provided by the Deere & Company whose sponsorship of this project is gratefully acknowledged.

Consequently, P_c decreases, resulting in a force imbalance on the swash plate (7 in Figure 6.1). The swash plate angle, θ, increases causing an increase in pump flow to the load orifice. This, in turn, increases P_s which is also sensed by the compensator orifice. The force imbalance on the compensator spool is reduced and a new equilibrium state is reached. The pressure drop across the load orifice is re-established and flow control is maintained.

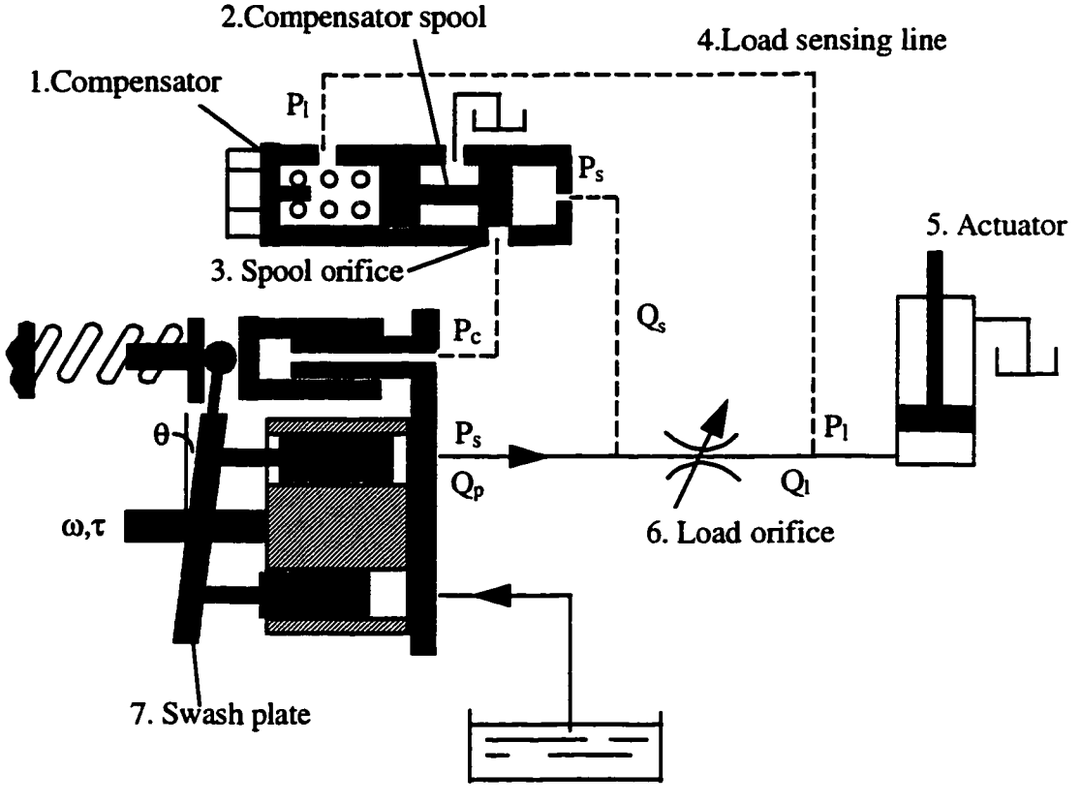


Figure 6.1 Schematic of a load sensing pump and resistive load system

6.3 Model Structure and Justification

The original model of the load sensing pump as an independent unit is illustrated by a causal block diagram as shown in Figure 6.2. There are three independent inputs to this model, shaft speed ω, pump output pressure P_s and load pressure P_l, and two

outputs, torque τ and pump output flow Q_p. The key to this form of the particular model is the independence of the inputs with respect to each other, as is required by the model verification. This means that, in experiments, a method of independently varying each input over a wide range of amplitude and frequencies must be devised.

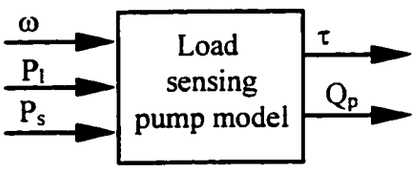


Figure 6.2 Causal block diagram of modeling a load sensing pump

Generally, the shaft speed is approximately constant (depending on the type of power drive to the pump shaft). For simplicity in this study, the input shaft speed is assumed constant and is not used as an input to this model[2]. (It should be noted that for a comprehensive model of the pump, several input shaft speeds should be considered as inputs.) P_l in the system is dependent upon P_s via the load orifice. However, if the P_l sensing line is removed to an external pressure source and the downstream side of the load orifice is set to tank, then P_l can be varied independent of P_s. This can be accomplished using a pressure control servo valve. It must be noted that the flow through the P_l sensing line is minimal which accommodates the use of a pressure servo valve.

In a working system, however, there is a strong dependency of P_s on P_l via Q_l. If the load orifice is replaced by a pressure servo valve to control P_s, then an independent P_s should be possible. However, preliminary experimental results clearly indicated that this was not the case with the equipment on hand because of excessive flow rate that the pressure control servo valve could not accommodate. Thus, the generation of an independent P_s became experimentally difficult.

[2] This assumption is substantiated by the fact that the pump is driven by a Crompton Parkinson 75 HP electric motor rated at about 1740 rpm at full load. The speed was found to be constant with the deviation of 0.67 % over the loading range examined.

In an attempt to make a compromise in the system to be modeled, a controlled resistance model was connected to the pump model as shown in the casual block diagram in Figure 6.3 (a). Q_l and I are the inputs to the load resistance model, and P_s is the output of this model and is now fedback to the pump model input. The causal block diagram shows that there are two independent inputs (P_l and I) and one output Q_l for the entire model. In many applications of load sensing systems, the resistance parameter (I) is held constant. Thus, to facilitate this study, I was adjusted to and then kept at a particular value to ensure that the swash plate of the pump operated over a reasonable range.

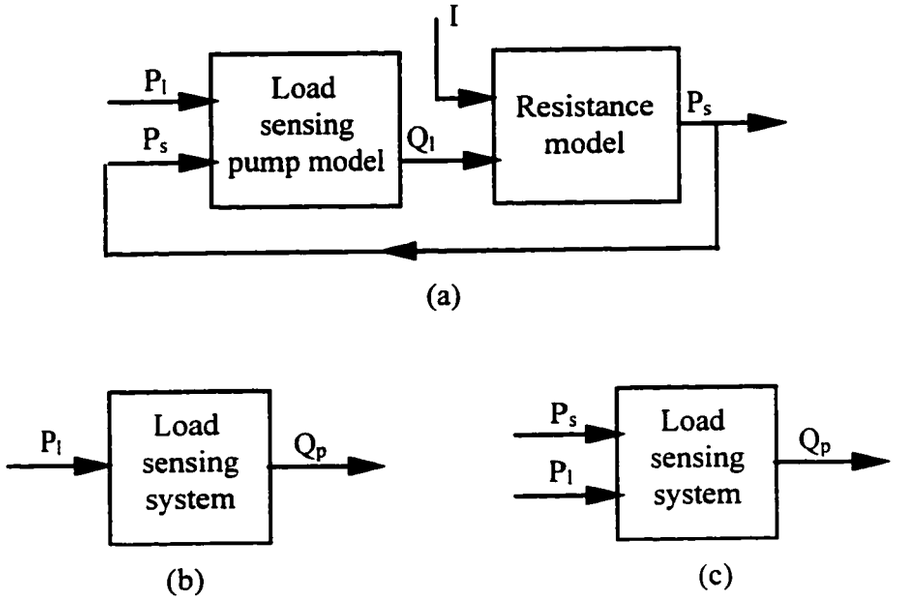


Figure 6.3 Causal block diagrams of the load sensing pump models:

(a) The load sensing pump model connected to a resistance model

(b) Modeling the SISO load sensing pump system

(c) Modeling the two-input single-output load sensing pump system

There were two scenarios to be examined. First, the entire model in Figure 6.3(a) was viewed as a single-input, single-output (SISO) model, and its causal block diagram is simplified as is illustrated in Figure 6.3(b). This form of the model actually provides a representation of pump dynamics combined with a particular load resistance. Secondly, if P_s and P_l are considered as measurable inputs and Q_l as the single output, the pump could then be modeled by a two-input, one-output model, as illustrated in Figure 6.3(c), but only at the operating loading condition defined by the dependency of P_s upon P_l via Q_l, i.e., the model is valid only for that particular controlling valve chosen. However, it should provide an approximation of the pump's own dynamics, albeit limited.

6.4 Experimental System

The experimental system is divided into a primary system and a secondary system, connected hydraulically through the pressure sensing line of P_l. The schematic circuit of the systems with appropriate instrumentation is shown in Figure 6.4. Details of all hydraulic components and associated electronics used in both systems are listed in Tables 6.1 and 6.2, respectively.

The load sensing pump was operated in the primary system and was driven by a Crompton Parkinson 75 HP electric motor rated at 1740 rpm with full load. The pump load compensator (reference to Figures 6.1 and 6.4) had two inputs, P_l and P_s. P_l was able to be controlled independent of P_s using an external pressure source provided by the secondary system (to be described latter). As discussed above, independent control of P_s with respect to P_l could not be achieved with available equipment in the laboratory. Consequently, a flow control valve was used to create a controlled resistance in the line. This accommodated the two model configurations in Figures 6.3(b) and 6.3(c). The pump flow was measured using two Ramapo drag type flow meters in parallel (two were needed because the pump flow rate of 19.5 GPM was greater than the maximum rating of each individual flow transducer).

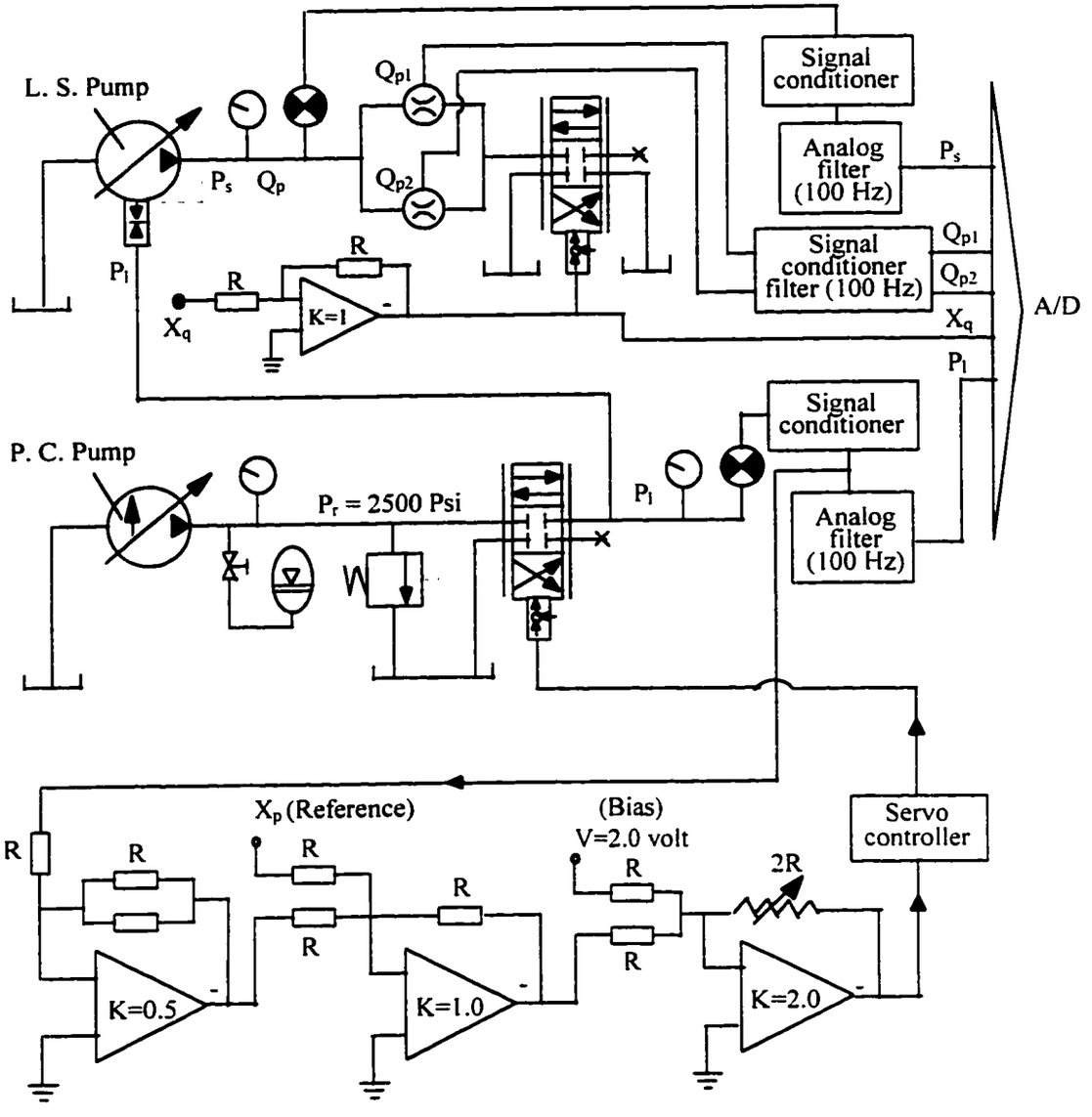


Figure 6.4 Identification experiment set up

Table 6.1 Primary system component list

Name	Amount	Model No.	Series No.	Others
Load sensing Pump	1	John Deere Part No.: AL75305	Not available	19.5 GPM
Flow control servo valve	1	Moog 62-115	1492	Rated flow: 15 GPM at 1000 psi
Flow transducer	2	V-.5-AOS5K 6-E	(1) 8710157 (2) 8710158	Range: 0 - 10 GPM ea.
2310 Signal conditioning amplifier	2	Not available	(1) 074422 (2) 054839	Used with flow transducers
Pressure transducer	1	Schaevitz type P1021-0005	113127	Range: 0-3500 psi
Low pass signal conditioner (300 Hz)	1	Not available	Not available	Used with pressure transducer
Rockland dual channel analog filter	1/2	852	Not available	One channel for Ps filtration at 100 Hz
Operational amplifier	1	(Engineering shop at U of S)	Not available	For current input to the flow control servo valve
Pressure gage	1	Not available	Not available	Range: 0 - 3000 psi
Temperature sensor	1	Type T thermocouple	Not available	For the primary system temperature
Scanning thermocouple thermometer	1	692-8000	Not available	Temperature indicator for both primary and secondary systems

Table 6.2 Secondary system component list

Name	Amount	Model No.	Series No.	Others
Pressure compensated Pump	1	PVB10RSY31 C11	Not available	
Pressure control servo valve	1	Moog 15 010	112	Rated flow: >14.3 GPM at 3000 psi
Moog D-C servo controller	1	Not available	Not available	For amplifying input signal to pressure control servo valve
Pressure relief valve	1	CT06F50	Not available	Range: 500 - 3000 psi
Accumulator	1	(Balder type)	Not available	
Needle valve	1	114286 A-2	6000 # 1/4	
Pressure transducer	1	Schaevitz type P1021-0005	113128	Range: 0-3500 psi
Low pass signal conditioner (300 Hz)	1	Not available	Not available	Used with pressure transducer
Rockland dual channel analog filter	1/2	852	Not available	One channel for P_l filtration at 100 Hz
Operational amplifier	3	604 (Burr-Brown)	Not available	For closed loop control of P_l (used with pressure control servo valve)
Pressure gage	2	Not available	Not available	Range: 0 - 3000 psi
Temperature sensor	1	Type T thermocouple	Not available	For the secondary system temperature

The function of the secondary system was to provide a controlled load pressure P_l using an external hydraulic power supply and a Moog pressure control servo valve. A pressure compensated pump in conjunction with an accumulator was used to create a constant supply pressure of 2500 psi. The presence of the relief valve was for safety purposes. Initial studies on the system using only a pressure servo valve indicated that control of P_l was possible only over a limited amplitude and frequency range. A proportional closed loop controller was then implemented and the bandwidth improved significantly. A frequency analysis of the closed loop servo system indicated that the control of P_l had been expanded to a bandwidth of approximately 100 Hz over reasonable amplitude. The obtained bandwidth was considered acceptable given that the maximum frequency of the pump to be modeled was approximately 50 Hz estimated from the preliminary tests. Three operational amplifiers were used to establish the closed control loop. A bias of 2.0 volt was required to set the operating range from 0 to 2500 psi. An additional Moog servo-controller was included in the loop to amplify the control signal to the pressure control servo valve.

In both systems, pressure and flow rate were measured with appropriate transducers and associated electronics. The pressure signals were passed through electronic signal conditioners and additional analog filters set at a cut-off frequency of 100 Hz. The two flow rate signals were conditioned using "2310 Signal Conditioning Amplifiers" which had filters set at 100 Hz. All signals were filtered at the same cut-off frequency of 100 Hz, and the phase shifts caused by the filters were found to be nearly the same for all measured signals. Thus, the effect of phase shifts caused by the filters need not to be considered. All measured signals of pressure and flow rates were then digitized using a data acquisition system described in the next section.

6.5 Data Acquisition Hardware and Software Implementation

Integral to the experimental study was the development of the software and the installation of the hardware for data acquisition. A schematic of the data acquisition

system used in this experiment is shown in Figure 6.5. The data acquisition system consisted of a 12 bit A/D (or D/A) converter, an IBM 386 computer and supporting software.

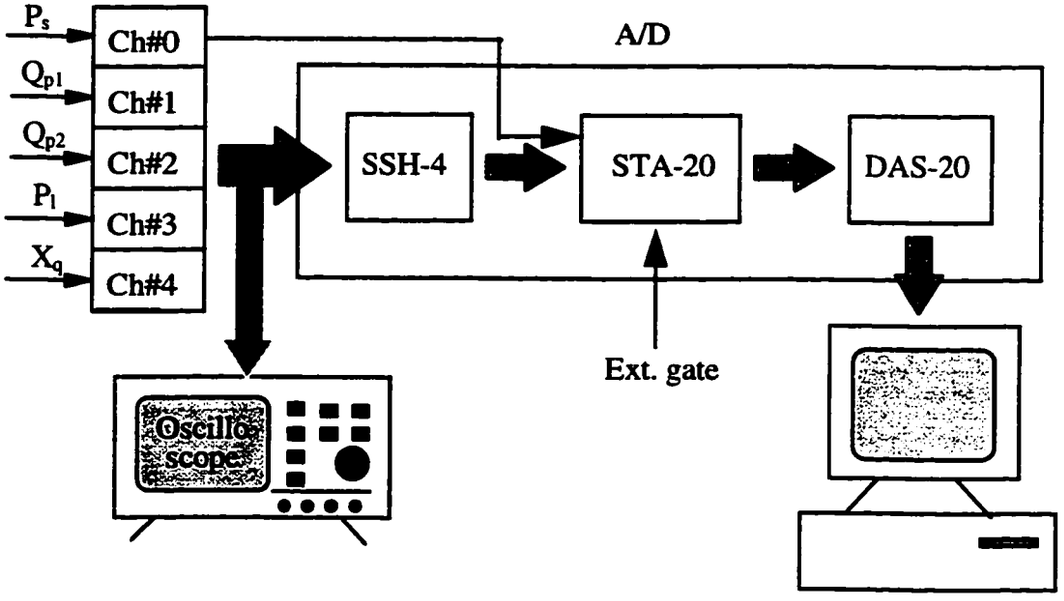


Figure 6.5 Data acquisition system

6.5.1 Hardware Installation

The data acquisition system comprised a D/A or A/D DAS-20 board (12 bit A/D & D/A Multifunction Data Acquisition board), a STA-20 board (Screw Terminal Accessory board), a SSH-4 (Simultaneous Sample & Hold for 4 input signals) board and connection cables (commercial products from Keithley Instruments, Inc.). The DAS-20 board had to be configured properly based on the user's selections before it was installed into the computer. There were four configuration options that could be selected using on-board switches. The four options were for (1) base address setting; (2) selection of eight differential or sixteen single-ended analog input channels (MUX configuration); (3) and (4) selection of analog output range of two channels (one switch for one

channel). The selection of analog input range was made through software. The board configuration for this study was as follows:

Base address:	300 Hex;
MUX configuration:	8-channel, differential input;
Analog output range for D/A0:	± 10 volt;
Analog output range for D/A1:	± 10 volt.

The resolution of this 12 bit board for a range of ± 10 volt is 0.004882 volt.

The pin of external gate #1 on the STA-20 board was connected to an external voltage source as a trigger source.

Originally, a C program was developed to perform a block scan of 5 channels with the controlled queue (specified sampling sequence) and sampling rate. That is, all of the five channels at each trigger were being scanned sequentially at the DAS-20's maximum sample rate, while the sampled data were transferred to the computer's memory via DMA (Direct Memory Access). This program was supported by several function calls including the call of Mode 27 that implemented "Perform N scans of the A/D Control Queue". However, initial studies indicated that as the A/D sampled 5 channels sequentially and evenly in one scan, a maximum time delay equivalent to 4 times the sampling periods existed. If all 5 channels were connected to the 60 Hz sinusoidal signal with a magnitude of 7.5 volt and sampled at 600 Hz, a maximum discrepancy of 0.1 volt between two channels resulted. Conceivably, the error would be larger if the signals vary faster. The error observed was considered to be significant for the measurement of system dynamics.

To improve the sampling accuracy, a special piece of hardware, SSH-4, was added to the system (see Figure 6.5). The SSH-4 performed four channel simultaneous sample and hold, and then transferred data to four of eight differential analog channels of the DAS-20 board. A SSH-4 board is normally used as a front end for the DAS-20 board. To accommodate a 5-channel measurement, both the SSH-4 and STA-20 were

used and connected in series, with four channel signals (Chan#1 - Chan#4) connected through the SSH-4, the one (Chan#0) connected directly to the STA-20. The SSH-4 was supplied with four bipolar, differential input channels (not selectable). Before the use of the SSH-4, the board needed to be configured properly for the range of each analog input with nine switch-selectable options, and for the output channel connections with jumper-selectable options. The configuration of the SSH-4 in this study was such that the range of all analog input channels were the same and set to 10 volt, and the four output channels of the board were linked to Channel #1 through Channel #4 of eight input channels to the board STA-20 which was then connected to the DAS-20 board.

To test the sampling accuracy for this arrangement, all 5 channels were connected to the same sinusoidal signals of 50 Hz and 75 Hz with a magnitude of 10 volt. The results showed that the sampling accuracy was improved in that the maximum discrepancy among the four channels through the SSH-4 was reduced to less than 0.01 volt (or 0.1% volt after normalized with respect to the maximum voltage magnitude), and the maximum discrepancy between channel #0 and any of the other four channels was 0.017 volt (or 0.17% volt after normalized). The sampling accuracy obtained was now considered to be acceptable.

6.5.2 Software Implementation

A commercial software package, STREAMER, was used to control the data acquisition with the use of the DAS-20 board. With reference to Figure 6.6, STREAMER allows DMA data transfer between the DAS-20 board and computer memory while transferring data between computer memory and the hard disk at the rate up to the maximum board speed. The amount of data that could be streamed automatically and continuously to the hard disk in an experiment was limited only by the actual space of the hard disk. This eliminated the memory problem that had limited the number of data transfer to 64 k bytes in one test with originally developed programs based on the PCF software package supplied with the DAS-20.

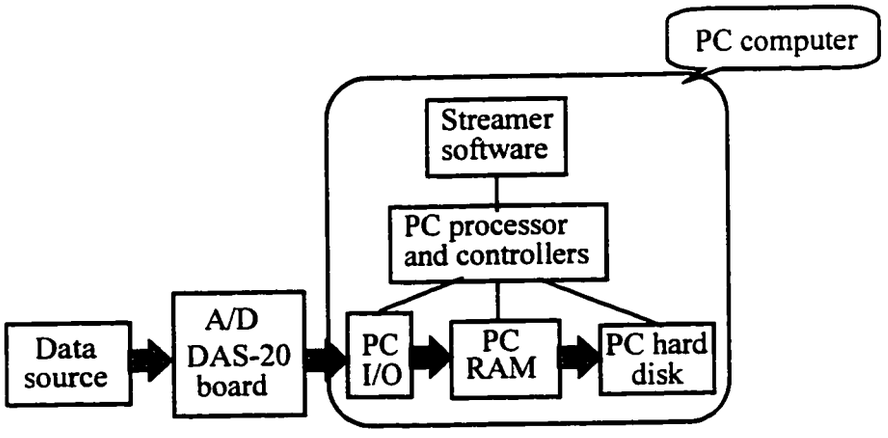


Figure 6.6 Block diagram of STREAMER-controlled system in data acquisition

A primary batch file, RUNSTR.BAT, was created to automate data conversion and transfer process as summarized in a flow chart in Figure 6.7. The procedure is described as follows:

(1) The first step was to defragment the hard disk if data on it was fragmentary. One of prerequisites for the use of STREAMER was that data files stored on the hard disk must be contiguous. STREAMER would not store data to a non-contiguous file. Whether or not there is a need for the defragmentation could be checked out when running MKFILE.EXE for creation of a binary data file in the next step. Defragmenting the hard disk could be performed by running a DOS command DEFRAG.

(2) The second step was to create a binary file with a specified size at root directory for data storage, using the MKFILE.EXE utility supplied with STREAMER package. MKFILE.EXE also checked the contiguity of data files. The file size must be specified in multiples of 1024 (1k) bytes.

(3) The third step was to check if the parameters required by STREAMER had been set up properly. STREAMER operates in either of two modes: Menu Driven mode or the BATCH mode. In either mode, the user specified arguments had to be supplied before initializing STREAMER. BATCH mode was used in this study. A batch file, STRMBAT.SRT, was created using the MKBATCH.EXE utility. Note that to avoid confusing this batch file with DOS batch files, STR, not BAT, was used as the file extension. This batch file contains a series of statements specifying the parameters for STREAMER used in this study, as listed in Table 6.3.

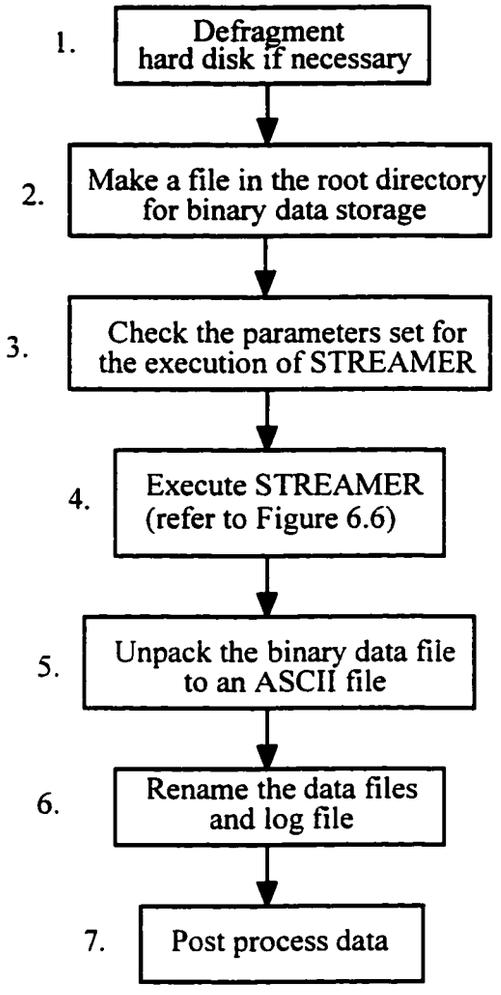


Figure 6.7 Flow chart of data acquisition procedure

Table 6.3 Parameter set up for STREAMER operations

Parameter	Specification
Board name/conversion mode	DAS20/ADC
Log file	C:\STREAM.LOG
Base address (Hex)	300
Scan mode	Block
Drive	C
Data file	STREAM.DAT
Sample rate (KHz)	0.250
Queue file	D20QUE.DAT
Clock signal source (CSRC)	INT
Trigger gate	EXT
Trigger signal source (TSRC)	INT
Interrupt level	2
DMA level	1

Two files needed to be created to support this batch file. One was the queue file, D20QUE.DAT, which specified the sequence at which five channels were scanned during A/D conversion, and the range of each channel (+10 volt for all channels in this study). The other was the log file, STREAM.LOG, that was used to record relevant information about the sampling operation, including all parameters specified, start and end time, total sampling time, actual number of samples, file size, and comments.

In the automated data transfer process, the content of the batch file was displayed on screen to allow the user to check for all parameters entered during the creation of the batch file.

(4) Execute STREAMER to initialize sampling operation. The data conversion was active as long as the gate signal remained a logic high. The operation continued until the specified data file was full.

(5) After the data transfer, the raw data stored in the binary file needed to be converted to an ASCII file to allow the data to be processed under text mode. A specially developed C based program, UPK.EXE, was executed to "unpack" the binary data file. This program enabled, with the arguments being key entered, the execution of UNPACK utility within the primary batch file. The arguments provided to UNPACK included the source file name (the binary file), destination file name, start and end numbers in the source file. Note: it was found that the start number must be specified in multiples of the total number of channels (5 in this study) for a reliable file conversion, and hence a zero start number was recommended.

(6) Upon the completion of a data acquisition process, three files were produced: a binary file for raw data, an ASCII file for the unpacked data and a log file for sampling information. All three files were renamed to avoid being overwritten by subsequent data transfer operations.

(7) All data obtained in the ASCII file were in digital form. A FORTRAN program, CONVT.FOR, was developed to normalize the data with respect to the maximum values. Calibration information on pressure and flow rate was used for the data conversion.

6.6 Calibration and Performances of Transducers

The quality of modeling any component using the neural network approach will certainly be affected by the reliability and accuracy of the instrumentation. Great care had been taken in calibrating the pressure and flow transducers and in understanding their performance.

The instrument for pressure measurement consisted of a pressure sensor and a signal conditioner that integrated electrical excitation, amplification and filtration (low pass). The pressure calibration was straight forward. A typical calibration curve is shown in Figure 6.8. The calibration data are listed Table A.1 in Appendix A. The transducer exhibited an excellent repeatability and linear relationship with a sensitivity of 1.0/300 (volt/psi). There was no hysteresis observed. The calibrated range was 0 - 3000 psi. The accuracy defined as the ratio of the maximum error to full scale output (10 volt) was 0.2% FSO (Full Scale Output)[3]. The mechanical natural frequency was specified as 5 KHz min which reflects a wide enough frequency bandwidth for measuring dynamic pressure. The pressure transducer was therefore considered to be accurate and reliable. Other relevant information for the pressure transducers can be found in Table 6.1 and Table 6.2.

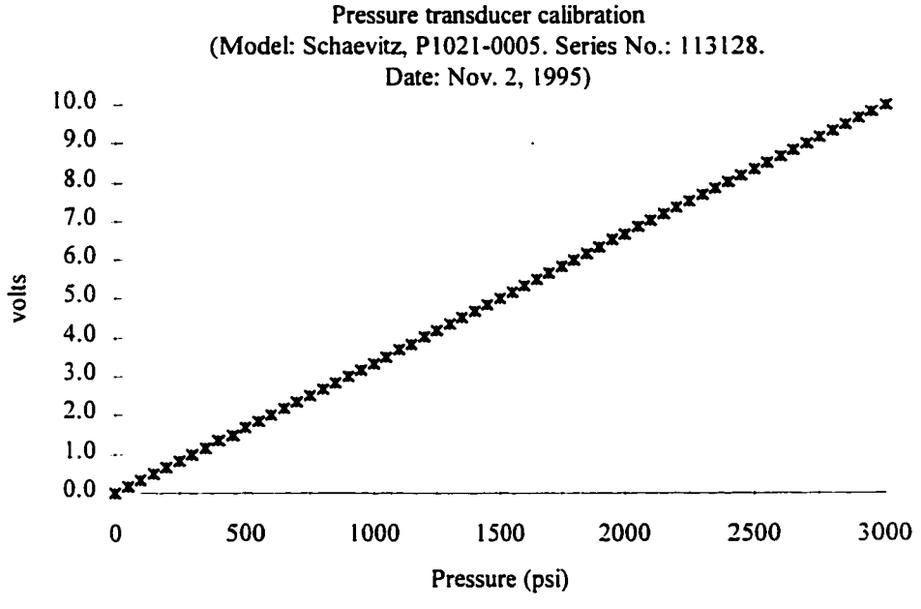


Figure 6.8 Pressure transducer calibration curve

[3] For the accuracy definition used here, the following is quoted from page 3 in "Industrial instrumentation" [Tyson, F. C., 1961]: "... when we speak of accuracy it will be in terms of full scale regardless of where the measurement is made on the scale. That is, if we measure a true 2 volts on a 0-100 voltmeter and find that our scale says 1 volts, we will understand this as accuracy of 1 per cent, not an error of 50 per cent. Of course we know that no intelligent student would ever try to measure 1 volt with a 0-100 voltmeter."

Particular attention had been paid to the calibration of the two flow transducers that were used in parallel for measuring large flow rates. The two flow meters were mounted in series into a specially controlled test system that provided an accessible method of obtaining static flow calibration. The flow meters were then calibrated independently, but against the same actual flow rates. The calibrated flow range was 0 - 10 GPM, and the full scale voltage output was 10 volt. Because the flow meters worked on the principle of drag force, the "raw" signals from the transducer exhibited a nonlinear power relationship between the actual flow and the measured signal. Preliminary tests showed that there was no observable hysteresis in calibration curves over a calibration cycle, and two tests indicated a good repeatability observed in the two calibration curves. A typical calibration record is summarized in Table B.1 (Appendix B). It had been observed that the second digit number after the decimal point in the voltmeter readings slightly fluctuated and drifted during the time interval over which the test system was calculating an average of actual flow rate. This phenomenon reflects the effect of temperature changes and stochastic nature of instantaneous flow rates. The output reading had to be taken based on an estimated average. This caused a calibration error in the voltage output and was estimated to be within $\pm$ .02 volt. It was also noticed that the calibration result was scattered in low flow rate range (less than 0.264 GPM) and therefore less reliable.

In order to use the calibration result to interpret flow rate measured from the experiment, curve fitting to the calibration result was necessary. Several fits were carried out using natural logarithms and three polynomials from 3rd to 6th order to minimize the deviation (or fitting error) from the actual flow rate. The fitting did not include the two smallest calibration points (refer to Table B.1 in Appendix B) as they were not reliable. A best fit equation was obtained among them based on the minimum deviation in terms of average and maximum fitting errors. The curve fitting results for both flow meters are summarized in Appendix C, and plotted against actual measurement in Figures 6.9 (a)

and 6.9 (b). When the output voltage was less than 0.01 volt, the corresponding flow rate was forced to be zero.

The overall measurement accuracy was then estimated based on a sum of A/D sampling error, calibration error and fitting error. The accuracy was specified in flow rate (GPM). Because of the nonlinear relationship, the calibration error of ± 0.02 volt reflected a maximum error at the smallest flow region due to the largest curve slope (refer to Figures 6.9(a) and 6.9(b)). The overall measurement accuracy for the two flow meters is estimated over the actual flow range approximately from 8 to 18 GPM (4 to 9 GPM at each flow meter) and is summarized in Table 6.4.

Table 6.4 Estimation of data measurement error

Estimated error	Flow Meter (Series No.: 8510157)	Flow Meter (Series No.: 8510158)
Maximum sampling error	0.14 % FSO	0.12 % FSO
Estimated maximum calibration error	0.27 % FSO	0.24% FSO
Maximum fitting error	0.38 % FSO	0.55 % FSO
Measurement error	± 0.79 % FSO	± 0.91 % FSO
Overall measurement error	± 1.70 % FSO	

Another important factor that would affect measurement accuracy is the variations of fluid temperature. However, the actual effect of the temperature on the overall measurement accuracy can be minimized by running all experiments at a pre-specified temperature. The final data accuracy in the flow rate primarily was dependent on the measurement accuracy calculated above, but also has affected by other factors

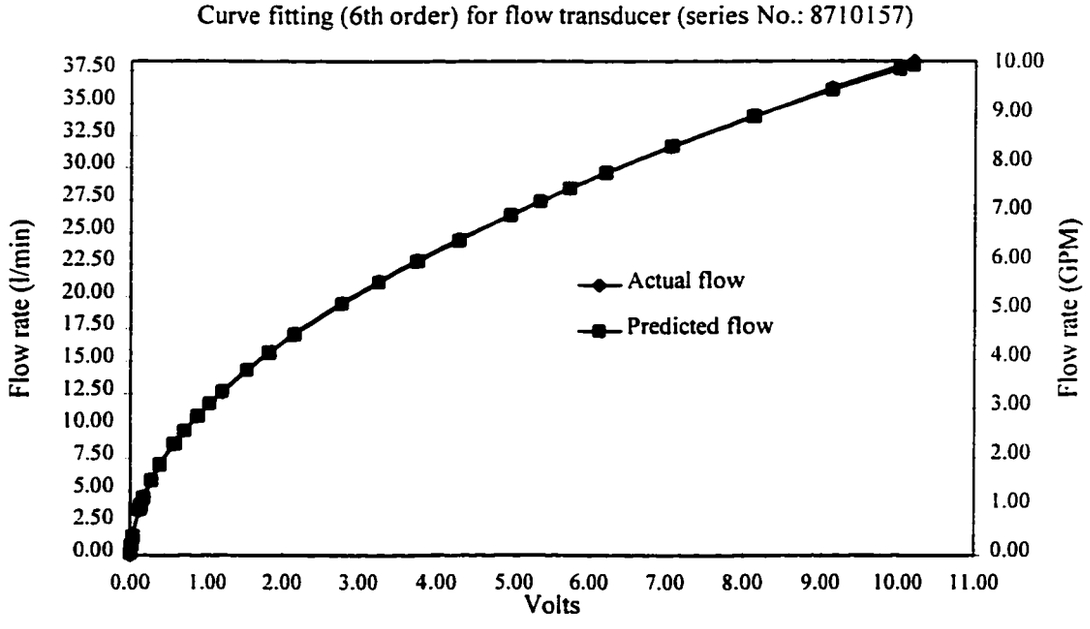


Figure 6.9 (a) For flow transducer with series No.: 8710157

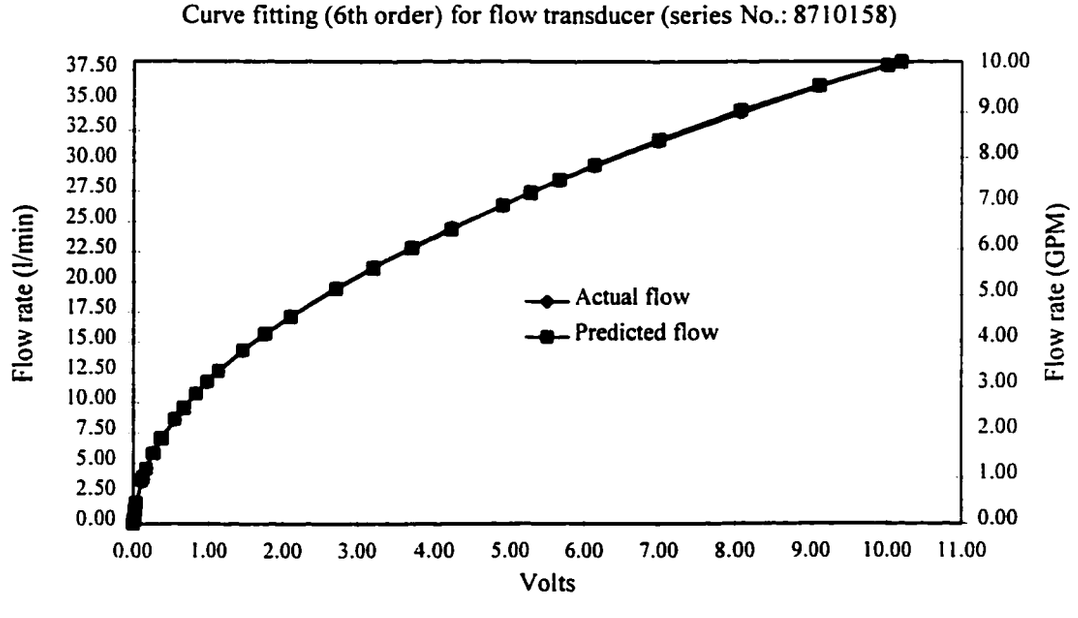


Figure 6.9 (b) For flow transducer with series No.: 8710158

Figure 6.9 Curve fitting for flow transducers

unpredictable from the testing environment and systems, such as minor erratic behavior in the flow valve.

The dynamic characteristics of the flow meters were examined in a specially constructed test stand. With typical inputs to the system, the flow to a linear actuator was measured by the calibrated flow meters and then compared to the actuator velocity measured by a linear velocity transducer. The results indicated that the flow transducers were able to provide reliable measurement of dynamic flow over the expected frequency range of this study.

6.7 Experimental Procedures

The experiment process consists of three stages: preliminary tests, input signal generation, and data collection.

6.7.1 Preliminary Test

The purpose of the preliminary test was to: (1) adjust the system to an adequate operating range; (2) ensure all hydraulic parts, instrumentation and data acquisition system functioned properly; (3) obtain knowledge on the dynamics of the load sensing pump that was to be identified. The information provided from preliminary tests would then be used to determine and to generate input signals.

Given the operating range of the load sensing pump at 0-19.5 GPM and 0-2500 psi, a desirable property for the experiment system was that the pump should be operated at a full range of flow over the full pump pressure range. The actual minimum pump pressure P_s was dictated by the load sensing compensator at a zero load pressure P_l, and was about 450 psi. The higher pump pressure P_s was created using a flow control servo valve in the line. The flow control valve was required to be set up properly in order to give a reasonable pump flow range via the pressure P_s created. The flow servo valve was

first partially closed and P_l (and hence P_s) increased until P_s was equal to 2500 psi. The valve opening was then adjusted until a full flow rate of 19.5 GPM was delivered by the pump. It must be noted that P_s could not increase beyond 2500 psi for that valve setting, because this was the maximum flow rate which the pump could deliver. For this valve setting, the minimum flow rate at the minimum pump pressure P_s (0 psi at P_l) was approximately 8 GPM. Thus, at steady state, the flow range for the one valve setting was 8 - 19.5 GPM for a pump pressure range of 450 - 2500 psi. It should be restated that for this feasibility study, the flow valve opening was fixed and not used as an input to the neural network model, the intention being to validate the neural network approach under a more controlled situation.

The dynamic response of the load pressure control system was examined through magnitude frequency response tests. The control signals were sinusoidal with a magnitude of 5 volt and frequencies up to 130 Hz. The responses of the load pressure P_l showed that no significant attenuation in magnitude of the pressure responses over the frequency range up to 100 Hz occurred, though some distortions in the wave forms were observed. The same experiment was conducted to test the bandwidth of the load sensing pump, and it was found that the pump flow response to load pressure P_l changes was negligible as the frequencies reach 50 - 60 Hz. Therefore the maximum bandwidth of the pump was ascertained to be about 50 Hz.

6.7.2 Input Signal Generation

As discussed in Chapter 2, an important requirement on the input signals for good generalization properties of the neural network, was that the input excitations contain a broad band of amplitude and frequencies with respect to the pump dynamics. As had been ascertained through the preliminary tests, the input signals should have a bandwidth of at least 50 Hz with P_l magnitude variations from 0-2050 psi, hence pump pressures from 450 - 2500 psi.

A pseudo-random signal was first considered. The signal was produced from a function generator with a 10 ms clock period and an infinite length. Its auto-spectrum and time history are shown in Figures 6.10 (a) and (b). Considering the first lobe, the maximum bandwidth of 100 Hz was sufficient for this experiment. However, the signal has only one magnitude component (± 2.0 volt). It was therefore necessary to devise a means to vary the amplitude of the signal without changing frequency components. The idea was to multiply the amplitude of the pseudo-random signal with a random signal uniformly distributed over 0-1.0 so that the resulting amplitude would vary randomly in a uniform distribution. The system illustrated in Figure 6.11 was used for this purpose. A programmed random number generator was implemented in a digital computer, and the random digital signals were converted to analog signals via the DAS-20 D/A converter. Each conversion was triggered by a positive slope of a pseudo-random signal from the function generator, and the random output level remained until next triggering. The programmed random signal generator was synchronized with the pseudo-random function generator. The two signals were then sent to the analog computer (Comedian GP-6) that had been configured to function as a multiplier. The signal, after multiplication (named "weighted pseudo-random signal" (WPR) for convenience), is shown in Figure 6.12 where the magnitude varies randomly with uniform distribution. Its frequency content remained the same as that shown in Figure 6.10 (a). This signal was recorded via the Cassette Data Recorder (TEACH XP-310) onto a video cassette at the maximum speed of 76 cm/sec. for later use.

In using the WPR signal, the tape was played back, and an operational amplifier was used to adjust the offset and the span. It was desirable for training to be conducted over the frequency band of the pump system and to be concentrated on the low frequencies, and a 50 Hz analog filter was thus employed to filter out high frequencies; this, however, did cause the magnitude distribution to be changed from a uniform to a Gaussian type of distribution. The filtered signal was used as the control input to the pressure control servo valve. Its frequency content is illustrated in a auto-spectrum plot in Figure 6.12.

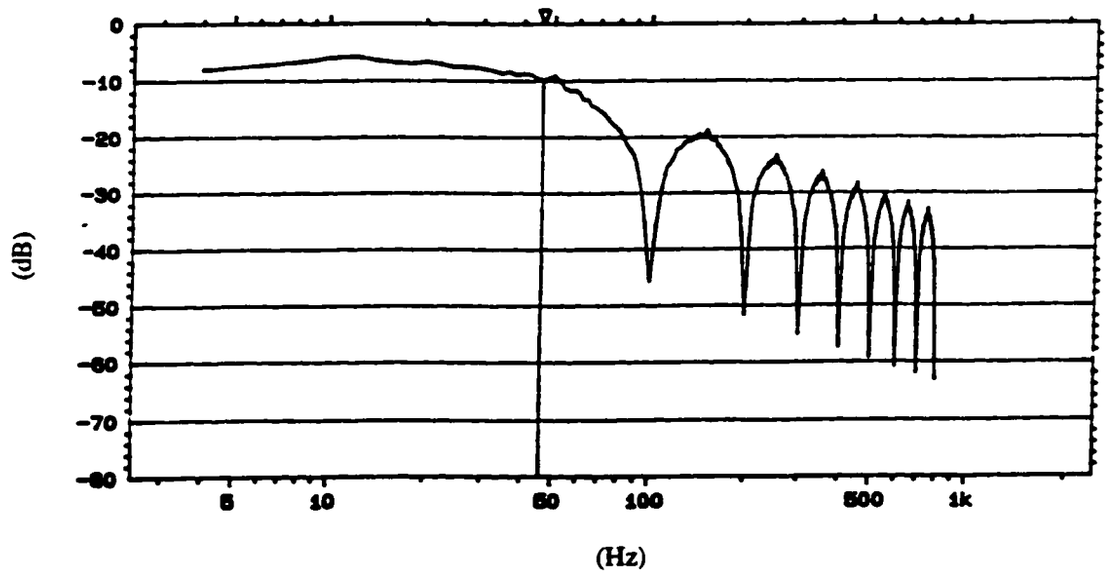


Figure 6.10 (a) Auto-spectrum

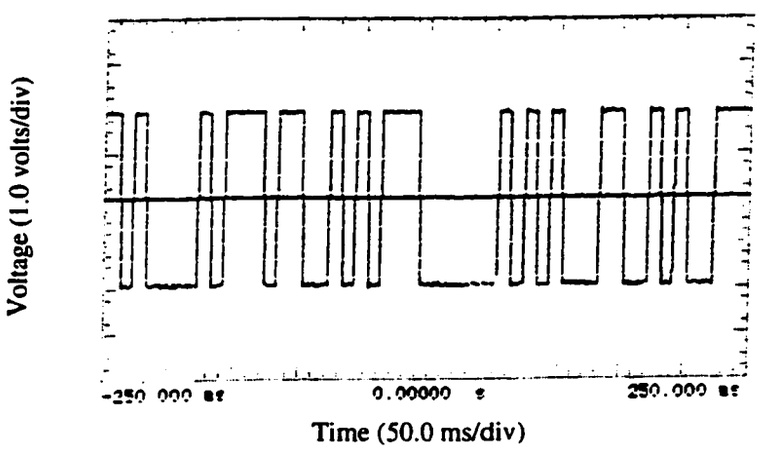


Figure 6.10 (b) Amplitude distribution

Figure 6.10 Auto-spectrum and amplitude distribution of the pseudo-random signal

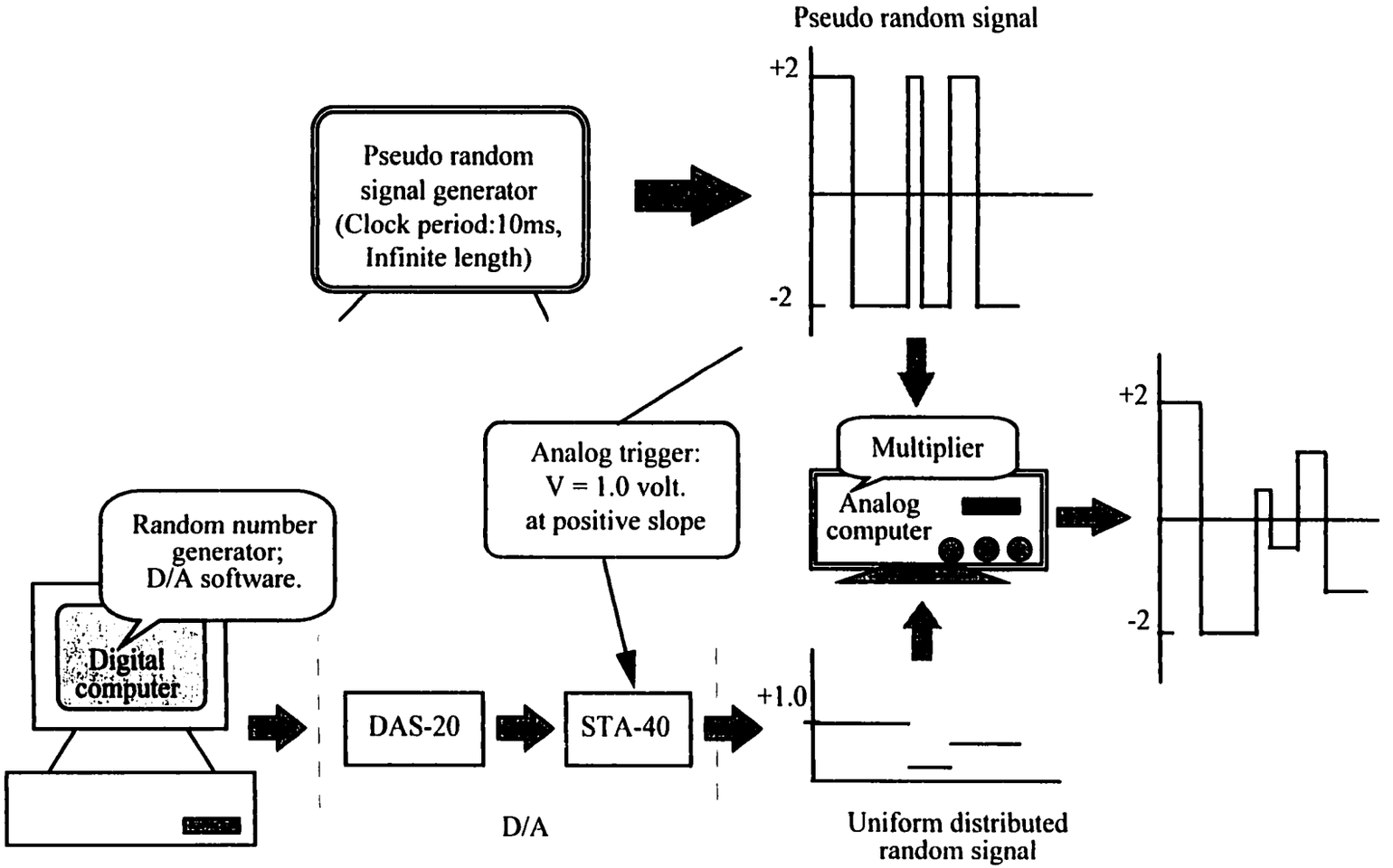


Figure 6.11 Input signal generation system

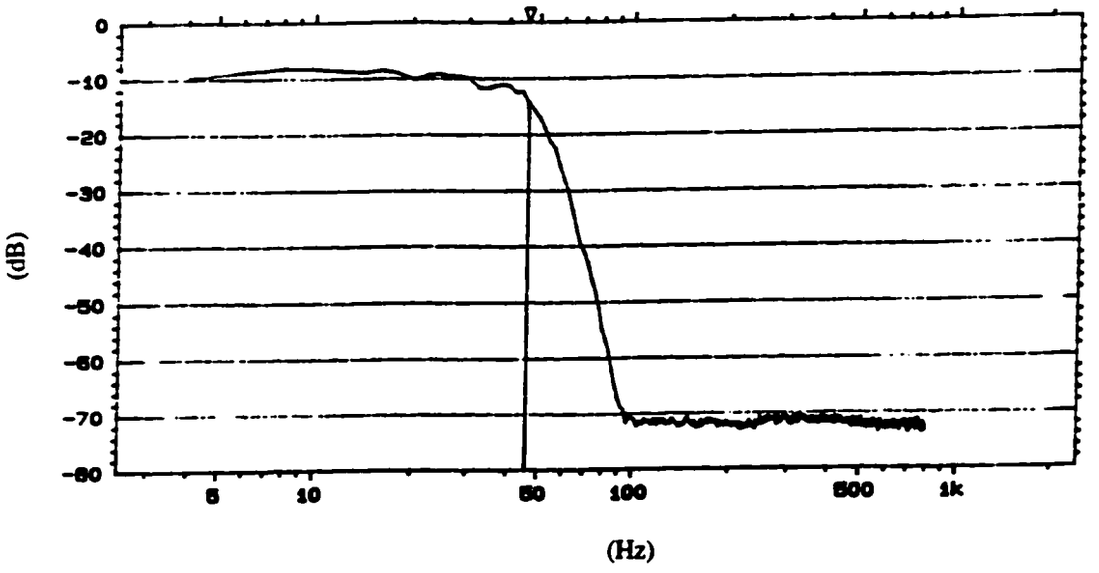


Figure 6.12 Auto-spectrum of the generated signal after 50 Hz filtration

6.7.3 Procedure of Test and Data Sample

Before the experimental tests, all appropriate electronic power supplies, amplifiers computers etc. were powered on. Two voltage generators needed to be set up. One provided a constant voltage (1.85 volt) to the flow control valve to create a fixed opening, as well as an external gate signal for A/D conversion triggering. The other supplied a constant bias to the closed loop pressure control system. The cooling system for the secondary system was turned on. The experiments were carried using the following procedures:

(1) Turn on the secondary pressure compensated pump whose deadhead was set to 2500 psi via the pump compensator;

(2) Set up the relief valve pressure to the maximum pressure of 2500 psi for safety purpose;

(3) Open a pre-charged accumulator via a needle valve to provide a controlled supply pressure and to absorb pressure spikes in the supply line;

(4) Turn on the load sensing pump;

(5) Run the whole system until the fluid temperatures for both systems reach 40 ± 5 ° C;

(6) Connect the output channel of the Cassette Data Recorder to the appropriate amplifier, and play back the video tape, exciting the pressure control system;

(7) Begin data collection by running the batch file RUNSTR.BAT;

(8) Disconnect the input signal immediately after completion of data sample;

(9) Turn off the systems: turn off the load sensing pump first; close accumulator; release relief valve pressure; and then turn off the secondary pressure compensated pump, and its cooling system as well.

Chapter 7
Experimental Modeling Results

7.1 Introduction

In this Chapter, the training and testing of a partially recursive neural network to model the dynamics of a load sensing pump using experimental data is presented. The data were obtained from testing the load sensing pump on the experimental system described in Chapter 6. Two different model configurations of the pump system are considered. The neural network is trained using a special type of random inputs produced by the signal generation system illustrated in Figure 6.11 of Chapter 6. The trained network is then tested against three different types of signals for model validity: sweeping sinusoidal inputs with frequencies from 0.001 to 60 Hz at a sweep rate of 0.22 decades/sec, step inputs from a triggered voltage generator, and random signals not used in the training. The modeling accuracy and generalization property of the neural network are examined and the effects of data noise and error accumulation through feedback paths on the modeling accuracy are discussed.

The results of this study show that the chosen neural network can be trained successfully using sufficient amount of experimental data to achieve the best training accuracy. The trained neural network is able to approximate the load sensing pump at transient and steady states with excellent generalization and satisfactory accuracy over the well trained operating range. The modeling accuracy is shown to be very dependent upon the training accuracy, which, in turn, is affected or limited by the experimental data quality.

7.2 Selection of Data Sampling Rate

It has been established through the simulation study in Chapter 4 that the data sampling rate is important for the discrete model of a dynamic system, because a high sampling rate could cause a significant error accumulation and hence a poorer modeling accuracy. The sampling rate should be chosen carefully such that it is not unnecessarily high causing significant error growth, but still high enough having sufficient discrete points to represent the transient state of the system response.

Two different rates of 600 Hz and 250 Hz were first attempted in training the neural network. It was observed that with 600 Hz sampled data, the trained neural network exhibited poorer accuracy when tested out in the model-driven mode because of more error accumulated through feedback paths (note that the network could be trained to achieve a better training accuracy in plant-driven mode). This result indeed verifies the modeling error analysis in Chapter 4. The rate of 250 Hz was adopted for all subsequent studies in the following sections.

7.3 Justification of Neural Network Morphology

The network that was used in this study consisted of 4 layers; an input layer, two hidden layers and an output layer. The neuron operations in each hidden layer could be programmed nonlinear or linear. The number of neurons in the input layer was dependent upon the number of inputs, and time delayed inputs and time delayed outputs. Only one neuron was in the output layer representing the pump flow rate. In choosing the morphology of the neural network, the decisions pertaining to the numbers of delayed inputs and delayed outputs, and the numbers of neurons in hidden layers had to be made. It is a desirable property that the neural model has the simplest structure, but still provide satisfactory model performance.

Some preliminary training experiments on the neural model structure were performed. The actual number of neurons in the first three layers were varied from experiment to experiment to observe the error function behavior. It was found that delaying the inputs did not improve the rate of training convergence, nor the training accuracy. Therefore, further use of time delayed inputs was no longer pursued. It was also observed that four output delays appeared to be the least number that still gave the best training accuracy. Increasing the number of output delays did not improve the accuracy but did increase the training time. The training experiments also showed that one nonlinear hidden layer was sufficient to provide the best training performance. The appropriate number of neurons in this layer was established to be between 15 and 20 for this particular study. Fifteen was finally chosen for the models. It should be pointed out that there was no apparent optimal size, and the actual morphology of the network was very application dependent. In addition, in order to examine the nonlinearity of the models, a neural network with no squashing function was investigated since it represented the simplest linear model structure. Its training results using the experimental data were considered fair but the testing results showed considerable error between the model output and the pump system output. Thus, nonlinear models should be used.

7.4 Establishment of Single-Input, Single-Output Model of the Load Sensing Pump System

The morphology of the neural network for single input single output (SISO) model of the load sensing pump system is shown in Figure 7.1. The input is load pressure P_l, and the output is the load sensing pump flow Q_p. The training information was 25000 - 30000 data pairs of P_l vs. Q_p. The autospectrum and time history of P_l are plotted in Figures 7.2 (a) and 7.2(b). Considering that the maximum frequency bandwidth of the pump system was about 50 Hz, the actual frequency bandwidth of P_l shown in Figure 7.2(a) was sufficient for the identification purpose. The amplitude variations of the input P_l shown in Figure 7.2(b) were 0-7 volts which satisfied the

requirement on the P_I variations (0.0-6.8 volts, corresponding to the pressure range of 0-2000 psi) to excite the full range of the pump flow rate (8 - 18 GPM). Before the training, the pressure and flow rate were normalized with respect to the maximum system pressure of 2500 psi, and the maximum pump flow rate of 19.5 GPM, respectively.

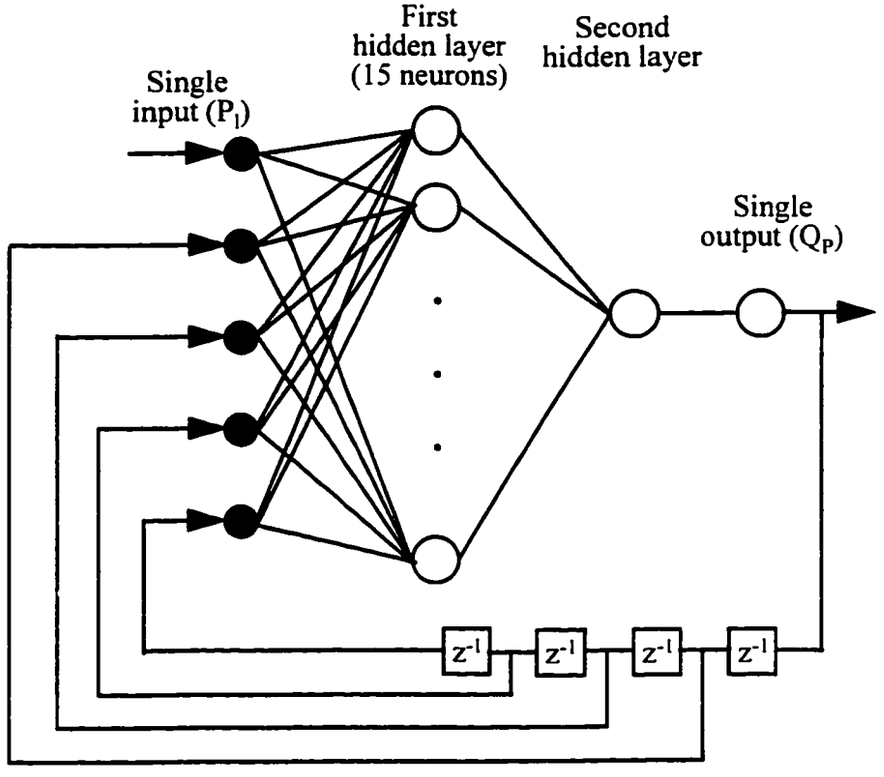


Figure 7.1 Morphology of SISO neural network model

The neural network was trained first using the plant-driven scheme as shown in Figure 4.2(a), i.e. the actual flow output from the pump was used as the feedback signal source to the input of the network. Batch training was performed. Each batch contained 1000 data pairs. The training was terminated when a pre-selected number of iterations were completed. This number was based on initial studies in which the error function

tended to level out over time. The error function was observed to monotonically reduce to the order of 8.0×10^{-5} for the last batch training.

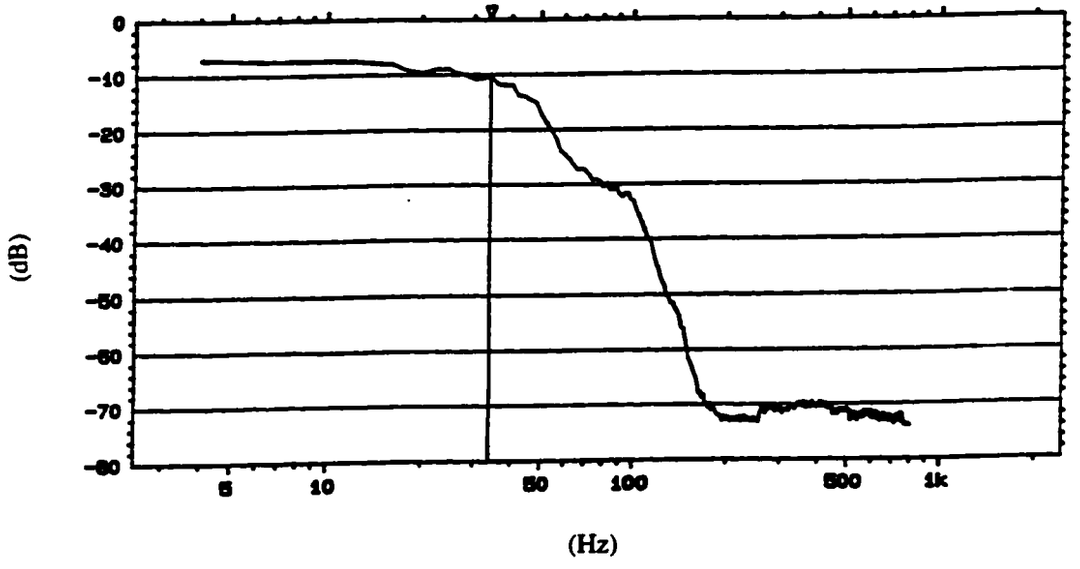


Figure 7.2 (a) Auto-spectrum of P_I

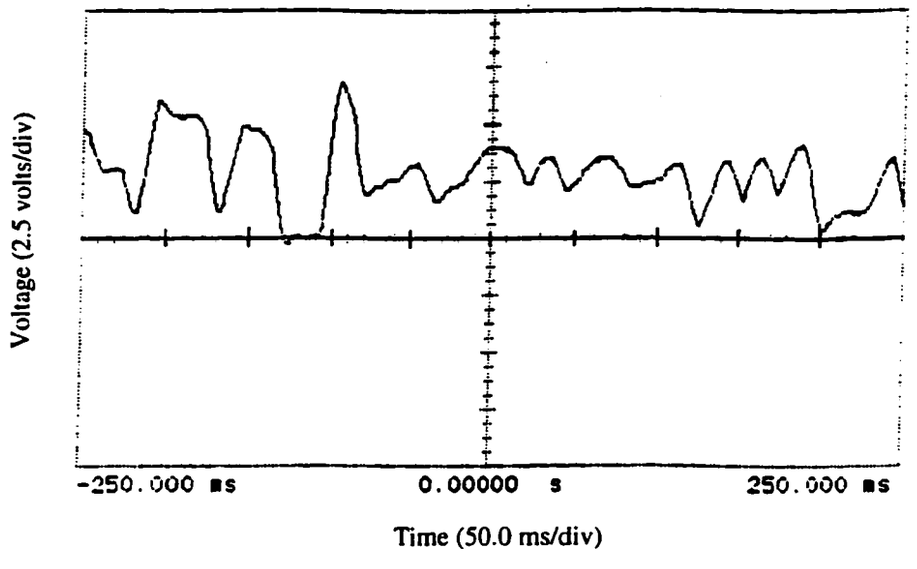


Figure 7.2 (b) Amplitude distribution of P_I

Figure 7.2 Frequency spectrum and amplitude distribution of input signal (P_I)

The trained network was then tested for model verification using the model-driven scheme shown in Figure 4.2(b) in which the model output (the predicted pump flow) was used as the feedback signal to the network input. It should be clear that, to verify the model as a simulator, the model had to be tested in the model-driven mode and with new data sufficiently covering the entire trained range. For comparison purpose, testing of the network using the plant-driven scheme was also performed in order to observe the error accumulation phenomena. All testing results in both schemes are presented in the following sections.

7.4.1 Model Validity Testing with Random Inputs

The trained model was subjected to a group of 2000 data pairs that were of a random type and were not used in the training. Typical testing results are illustrated in Figures 7.3(a) and 7.3(b) for the plant-driven mode and Figures 7.4(a) and 7.4(b) for the model-driven mode. Only a portion of the 200 data pairs are shown in each plot for clarity purpose. The corresponding NRMS values are summarized in Table 7.1.

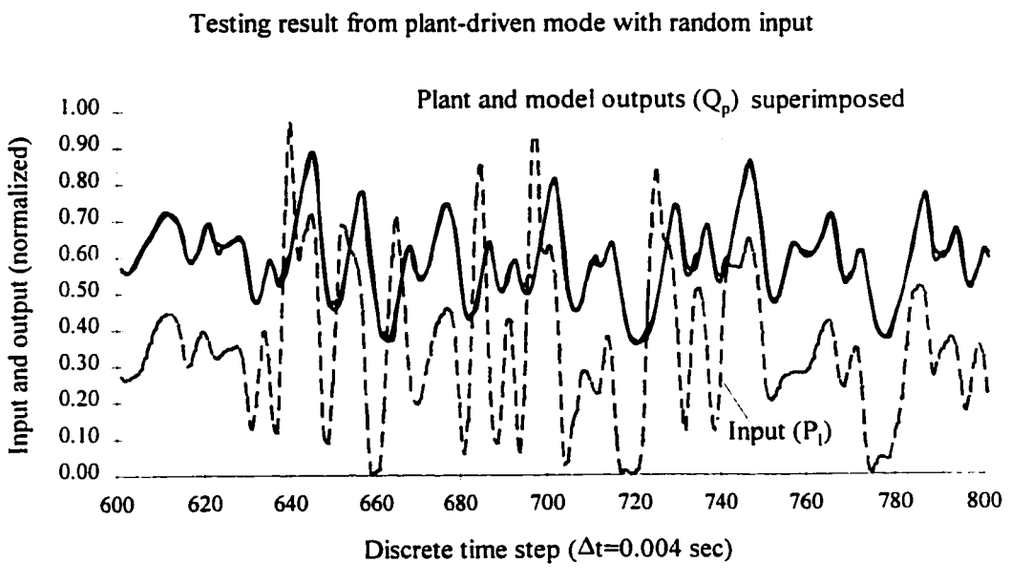


Figure 7.3(a) Plant-driven testing result using random input (time step: 600-800)

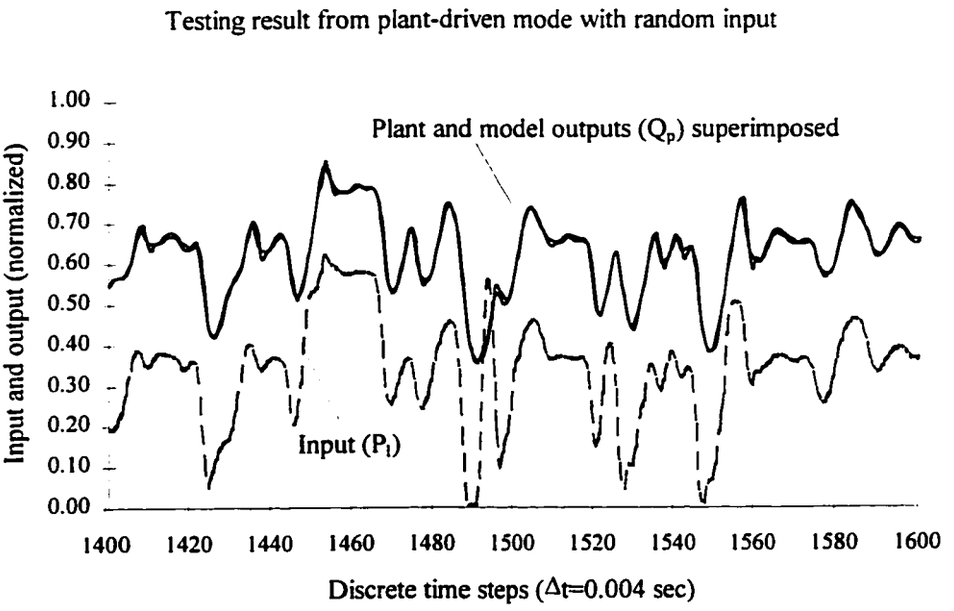


Figure 7.3(b) Plant-driven testing result using random input (time step: 1400-1600)

Figure 7.3 Plant-driven testing results using random input

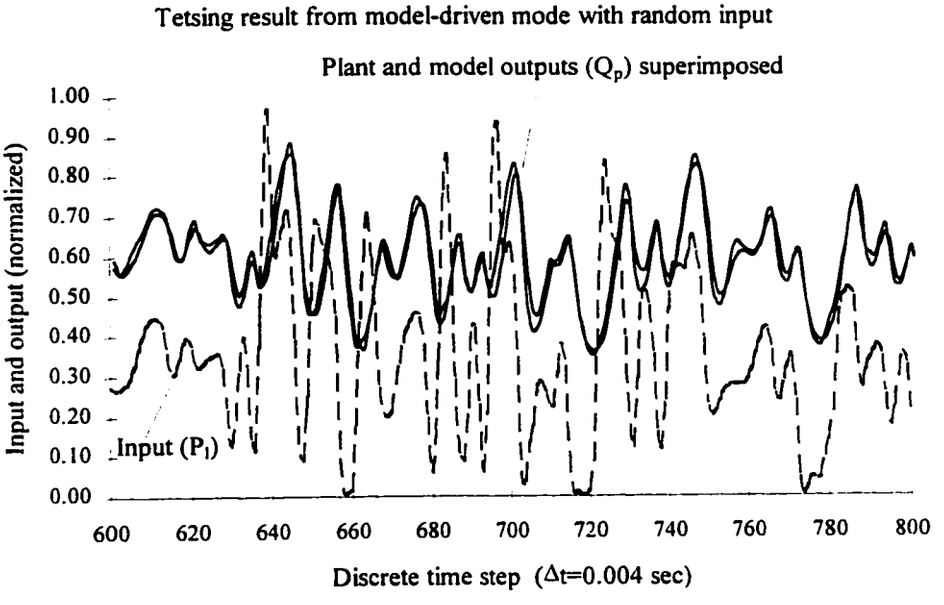


Figure 7.4(a) Model-driven testing result using random input (time step: 600-800)

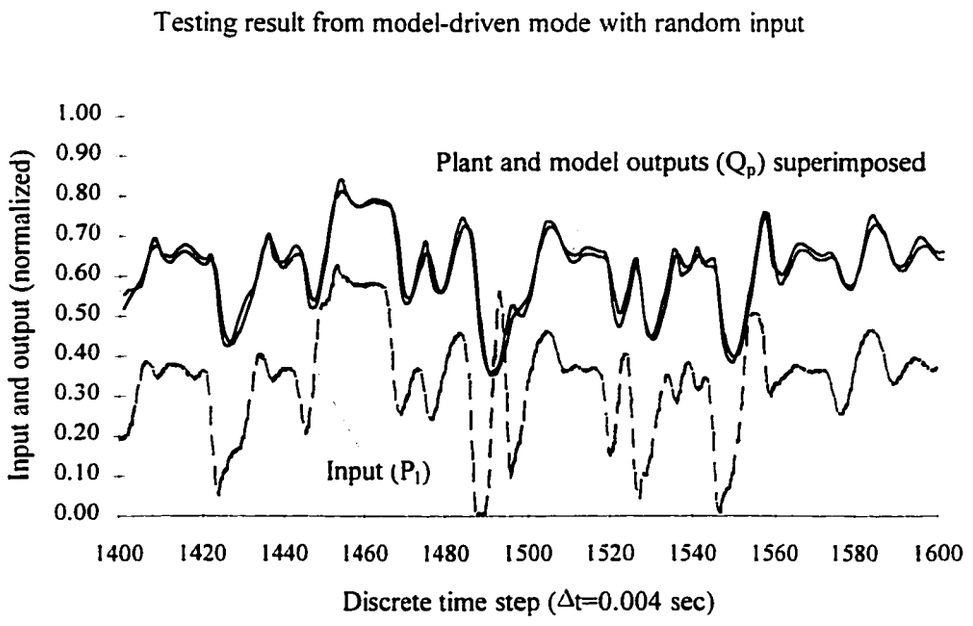


Figure 7.4(b) Model-driven testing result using random input (time step: 1400-1600)

Figure 7.4 Model-driven testing results using random input

Table 7.1 SISO model testing results with random input

Input type	NRMS (plant-driven)	NRMS (model-driven)
50 Hz random	0.014157	0.029644

The accuracy in the model-driven testing decreased by an approximate factor of 2 in NRMS values, compared to the plant-driven testing. However, the normalized RMS value of less than 3% in the model-driven scheme is still considered very satisfactory as the results show excellent generalization and modeling accuracy for the model as a simulator over the tested region.

7.4.2 Model Validity Testing with Swept Sine Wave Input

The swept sine wave inputs were generated using a Bruel & Kjær Signal Analyzer Unit Type 2035. The input and output responses were obtained in 8 groups. The first 4 groups had a sweep range of frequencies from 0.001 Hz to 1.0 Hz, but with varying amplitude. The remaining 4 groups had a sweep range from 1.0 Hz to 60 Hz with the same magnitude variations as in the first 4 groups. Eight tests were carried out in the plant-driven and model-driven modes, respectively. The NRMS values are summarized in Table 7.2.

Table 7.2 SISO model testing results with swept sine wave inputs

Input type: swept sine wave (test#1 - test#4: 0.001-1.0 Hz test#5 - test#8: 1.0 - 60 Hz)	NRMS (plant-driven)	NRMS (model-driven)
test#1	0.00972	0.02017
test#2	0.00705	0.01481
test#3	0.00510	0.01083
test#4	0.00437	0.00907
test#5	0.01166	0.01918
test#6	0.01268	0.02096
test#7	0.01164	0.02092
test#8	0.00879	0.01574

Typical results from test#4 and test#8 are presented in Figures 7.5 and 7.6 for demonstration purposes. The two plots in Figures 7.5(a) and (b) show the testing results from the test#4 over a complete sweeping cycle from 0.001 to 1.0 Hz in the plant-driven and model-driven modes, respectively. Only a portion of testing results from test#8 are plotted in Figures 7.6(a) - (d) for clarity purpose. The plots in Figures 7.6(a) and (b) are the plant-driven testing results over the swept inputs of 1-10 Hz and 25-40 Hz,

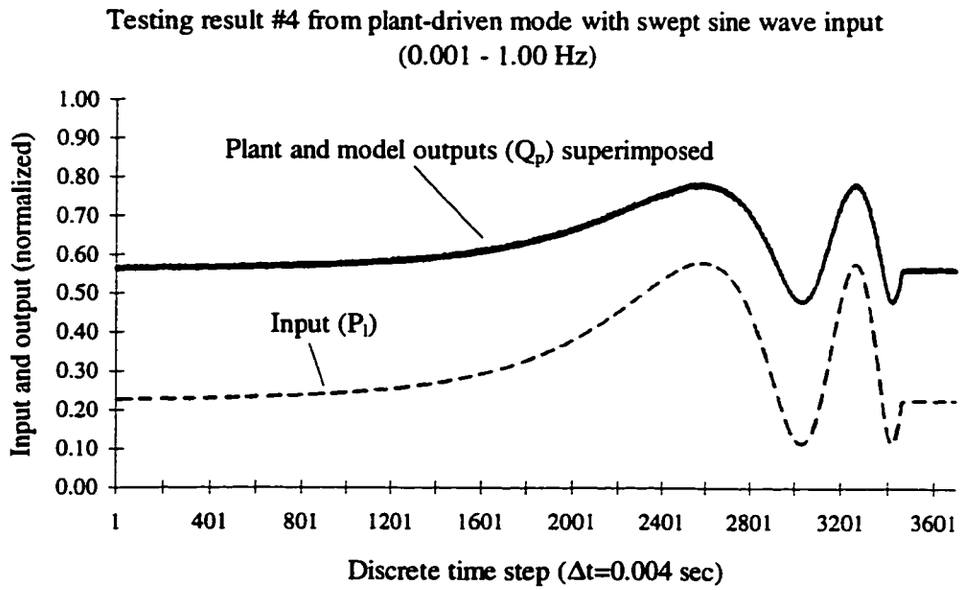


Figure 7.5(a) Plant-driven testing result

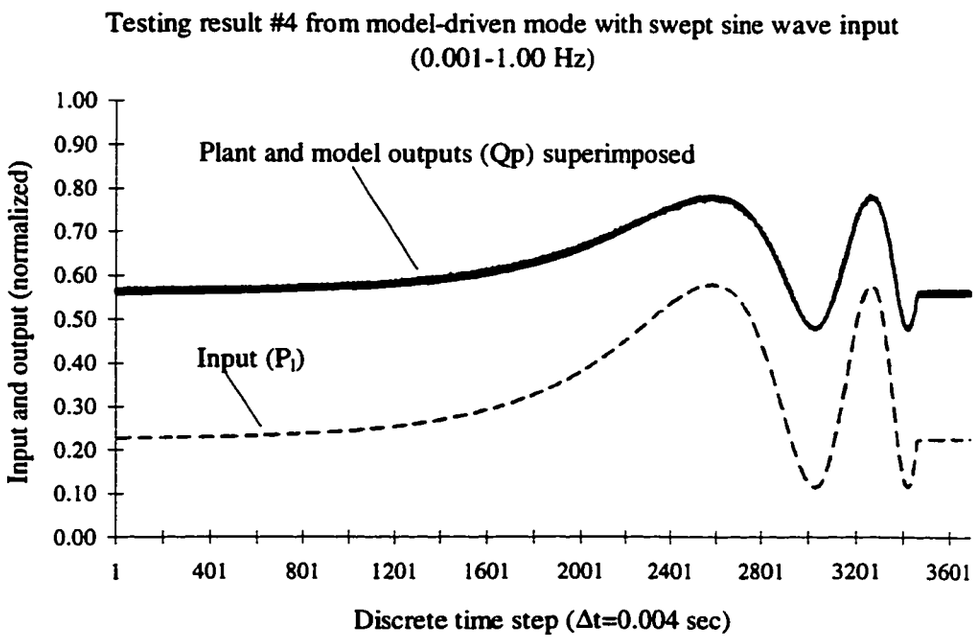


Figure 7.5(b) Model-driven testing result

Figure 7.5 Model testing results using swept sine wave input (0.001 Hz-1.0 Hz)

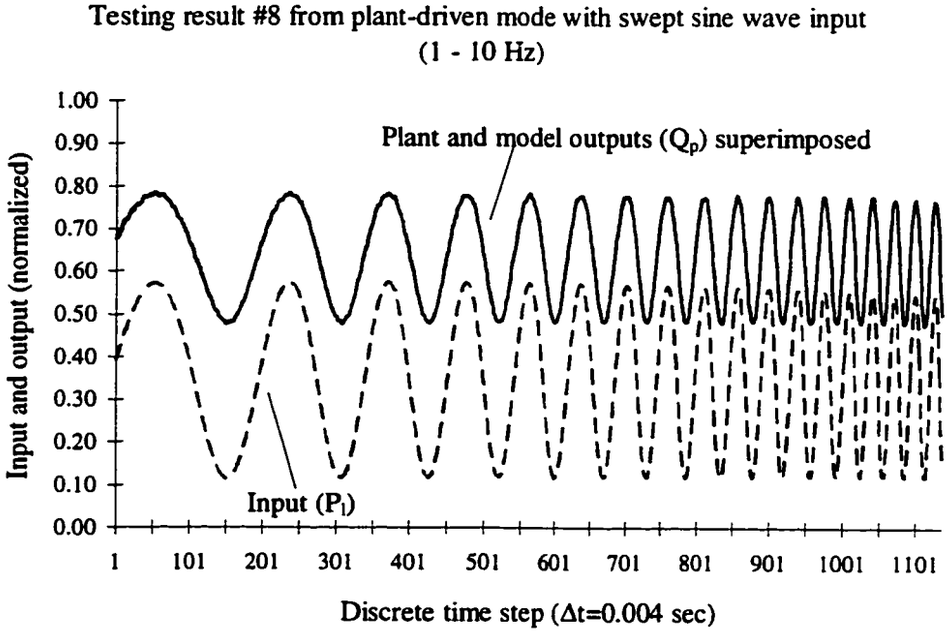


Figure 7.6(a) Plant-driven testing result (1.0 Hz - 10 Hz)

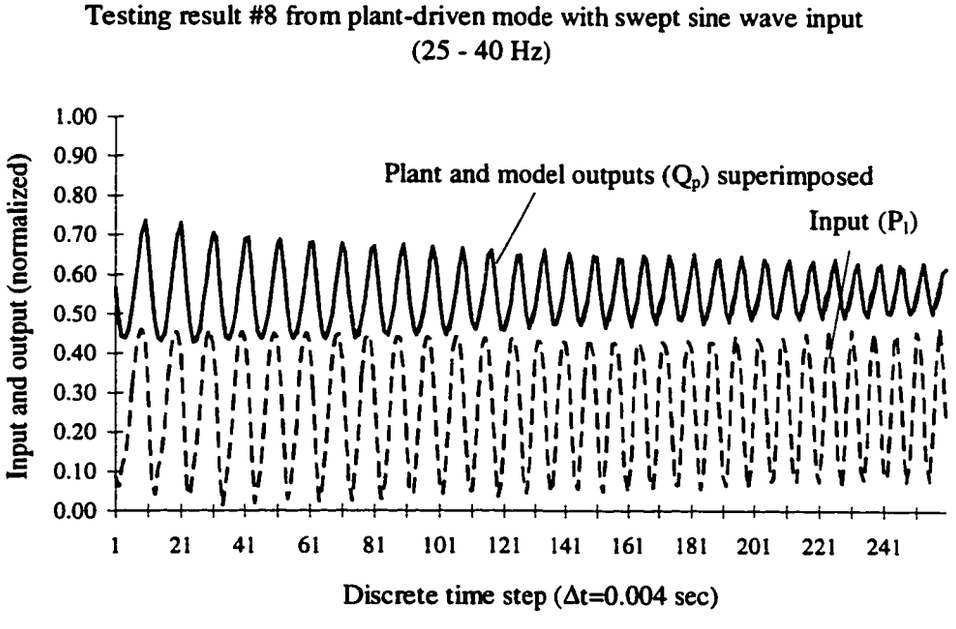


Figure 7.6(b) Plant-driven testing result (25 Hz - 40 Hz)

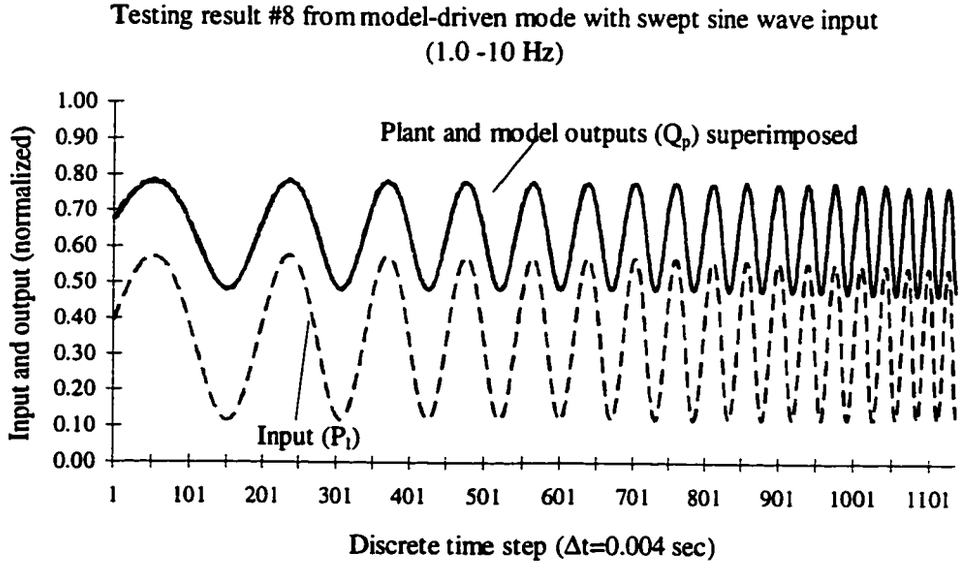


Figure 7.6(c) Model-driven testing result (1.0 Hz - 10 Hz)

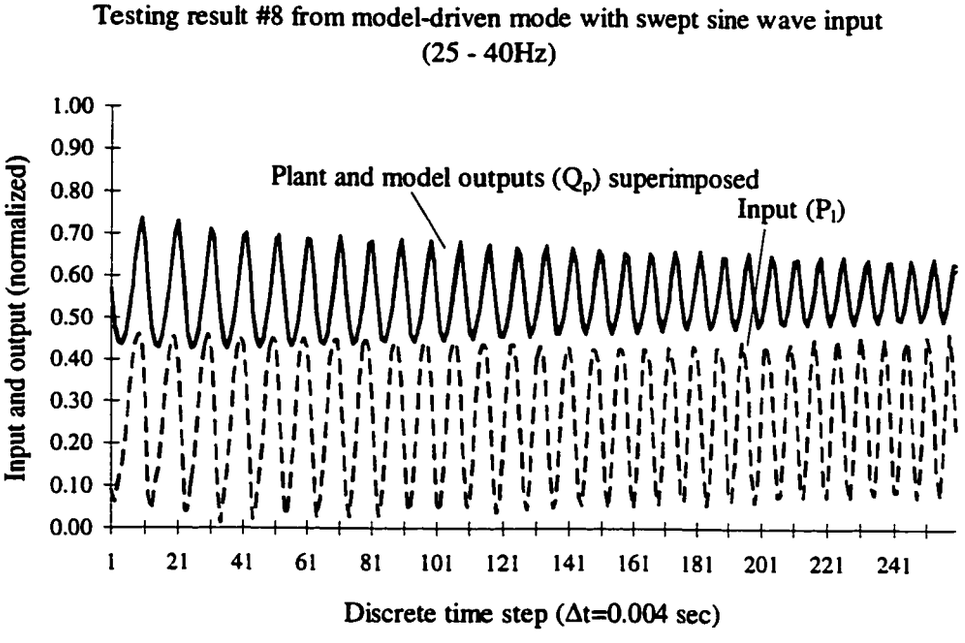


Figure 7.6(d) Model-driven testing result (25-40 Hz)

Figure 7.6 Model testing result #8 using swept sine wave input

respectively, and the plots in Figures 7.6(c) and (d) are for the same swept inputs, but in the model-driven mode.

The use of the sinusoidal signal swept from 0.001 to 60 Hz essentially tested the model frequency response property over the 60 Hz bandwidth. The swept sinusoidal testing information can be translated into an approximate frequency response plot. A typical magnitude frequency response plot is shown in Figure 7.7. The plant-driven testing result over the swept inputs of 40-60 Hz is plotted in Figure 7.8 for comparison purpose.

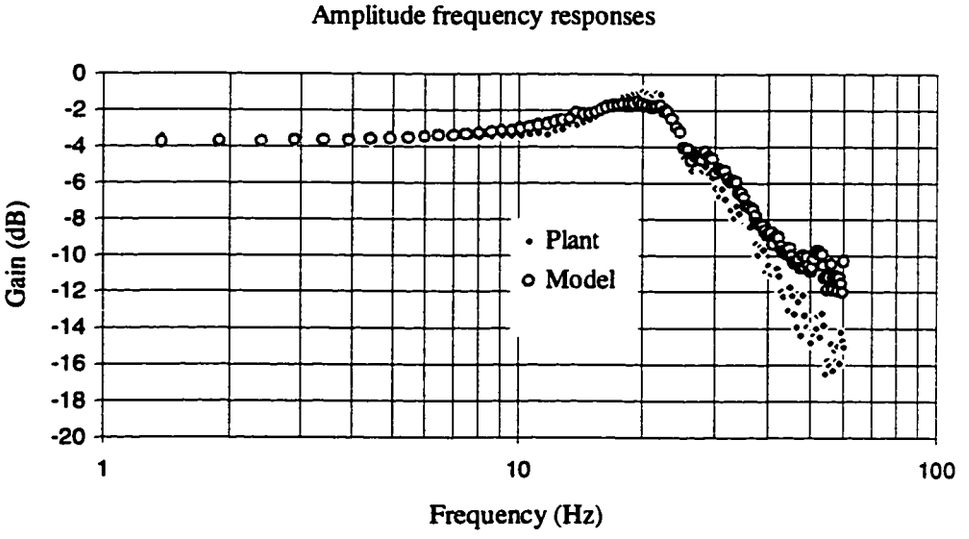


Figure 7.7 Amplitude frequency response comparison between the load sensing pump and its neural model

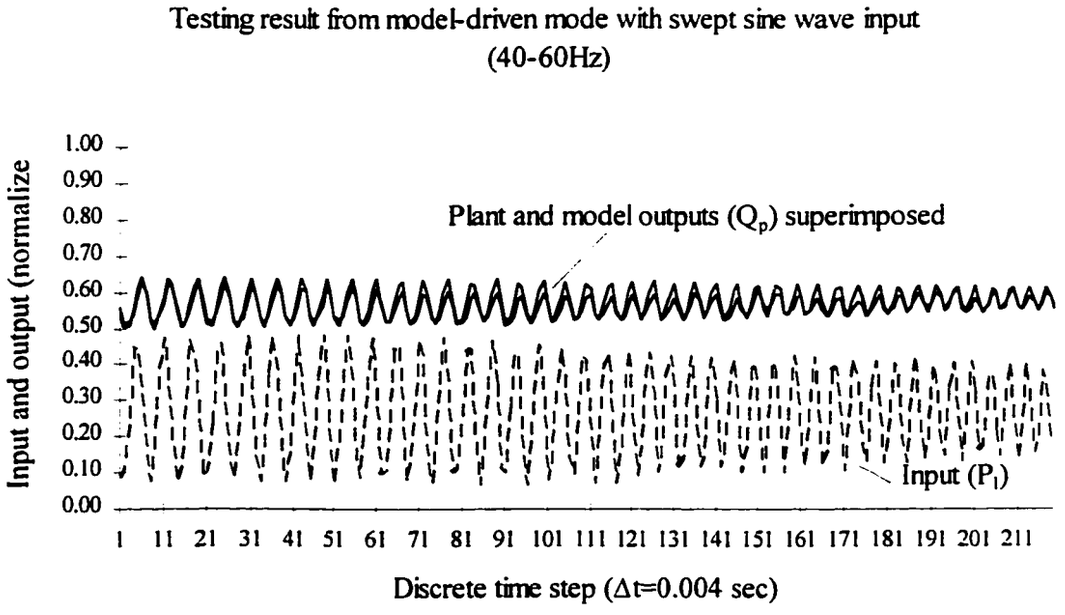


Figure 7.8 Model-driven testing result with swept sine wave input (40-60 Hz)

Upon the observation of Figure 7.7, it is very noticeable that the magnitude ratios for the plant and the neural model are very close up to approximately 30-40 Hz. At higher frequencies, the magnitude ratios of the plant and the model differ. The same trend is observed in Figure 7.8. Overall, the trained neural model is able to approximate the pump system with good agreement (maximum NRMS of 2.1%). The error accumulation, however, can be observed by examining Table 7.2.

7.4.3 Model Validity Testing with Step Inputs

A pure step input in the load pressure P_l was difficult or, indeed, impossible to be obtained in the real system. The actual step inputs P_l used in this study could only be considered as "quasi-step" inputs. The P_l quasi-step signal was obtained using a voltage generator. A step voltage signal was triggered manually and sent to the load pressure control valve. The corresponding P_l, Q_p and P_s were recorded through the data acquisition system. By appropriately setting the voltage level, variable step magnitudes

of P_l could be obtained. Ten tests were performed using varied step inputs to ensure a sufficient coverage of the trained range. The corresponding NRMS values for the 10 tests in both the plant-driven and model-driven modes were calculated based on a sum of errors in both transient and steady states, and are listed in Table 7.3. It should be pointed out that the use of a step input for the model testing imposed a most restrictive testing condition, because the rising edge of the step signal contained very rich high frequency components, which were often not covered by the training signal with limited frequency bandwidth.

Table 7.3 SISO model testing results with step inputs

Input type (quasi-step input)	Input magnitude P_l (steady state values[1])	NRMS (plant-driven)	NRMS (model-driven)
test#1	0.109	0.00753	0.02812
test#2	0.146	0.01069	0.02905
test#3	0.216	0.01274	0.02972
test#4	0.279	0.01177	0.02931
test#5	0.349	0.01106	0.02550
test#6	0.411	0.01065	0.02497
test#7	0.459	0.01532	0.04229
test#8	0.523	0.01240	0.02699
test#9	0.577	0.00965	0.02413
test#10	0.634	0.00767	0.02388

Three typical results from tests #2, #6 and #10 are presented in Figures 7.9(a) - (c) for the plant-driven mode, and Figures 7.10(a) - (c) for the model-driven mode. Ten testing results in the model-driven showed that the model was able to approximate the pump plant output with close agreement. The accuracy was very satisfactory for the first 6 tests where the step response magnitudes were less than 0.73 (normalized value), but

[1] P_l is normalized with respect to the maximum P_s of 2500 psi for the operating range considered in this thesis. The maximum normalized value of P_l is 0.8 corresponding to the maximum actual P_l of 2000 psi.

decreased as the step size increased in tests #7- #10. (see the example of test#10 in Figure 7.10(c)). This was attributed to the fact that the accuracy in plant-driven mode also slightly decreased in the larger step response region (0.7-0.9) in tests #7 - #10. As shown in Figure 7.9(c), this decrease in accuracy was amplified in the model-driven mode, because of the error accumulated through four feedback paths. Such accuracy deterioration between the two modes can be clearly observed by a comparison of the plots in Figure 7.9(c) and Figure 7.10(c).

In summary, all testing results on the SISO model using different types of new data have consistently shown very satisfactory accuracy over most of the trained range, and hence established the model validity as a simulator. The causes for the local poorer accuracy in plant-driven mode will be discussed in later section of this Chapter.

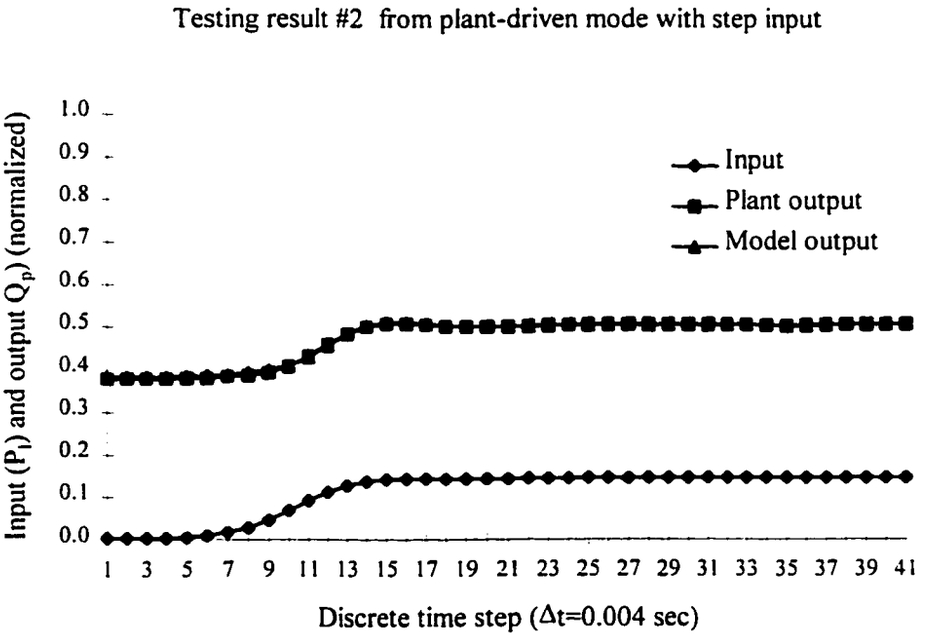


Figure 7.9(a) Plant-driven testing result #2

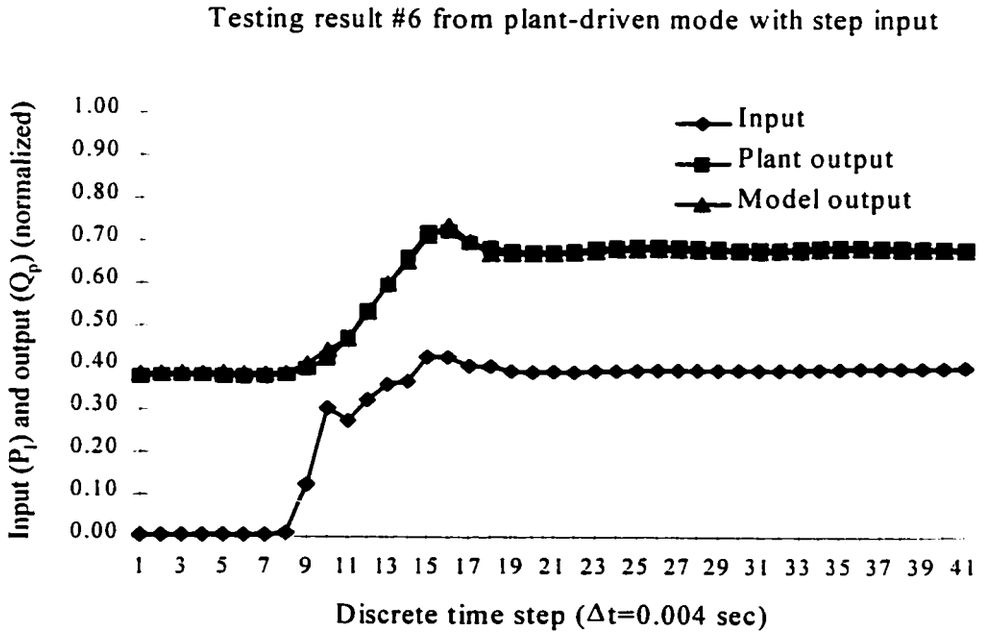


Figure 7.9(b) Plant-driven testing result #6

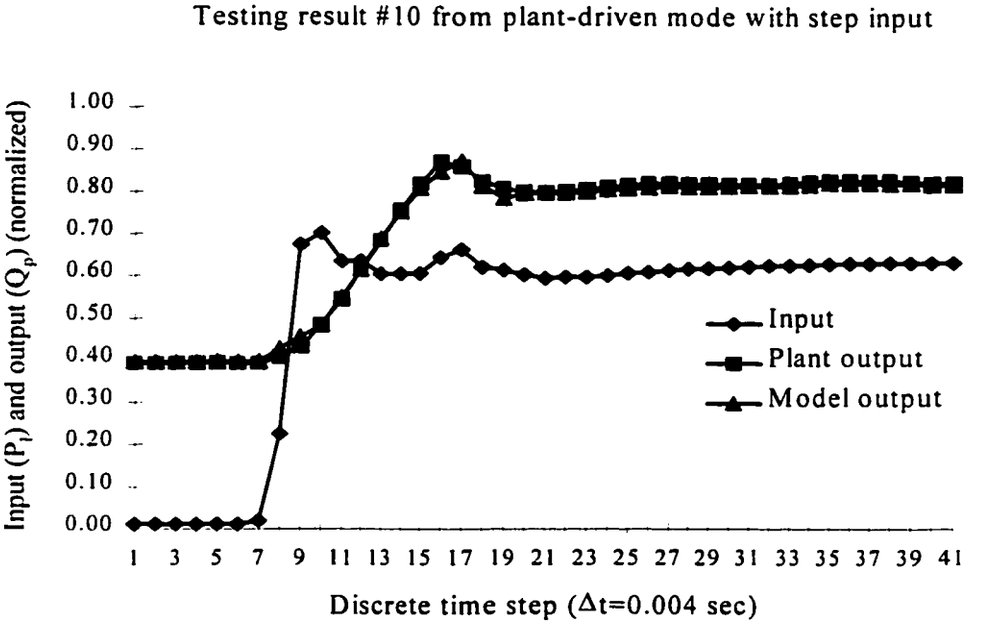


Figure 7.9(c) Plant-driven testing result #10

Figures 7.9 Plant-driven testing results using step inputs

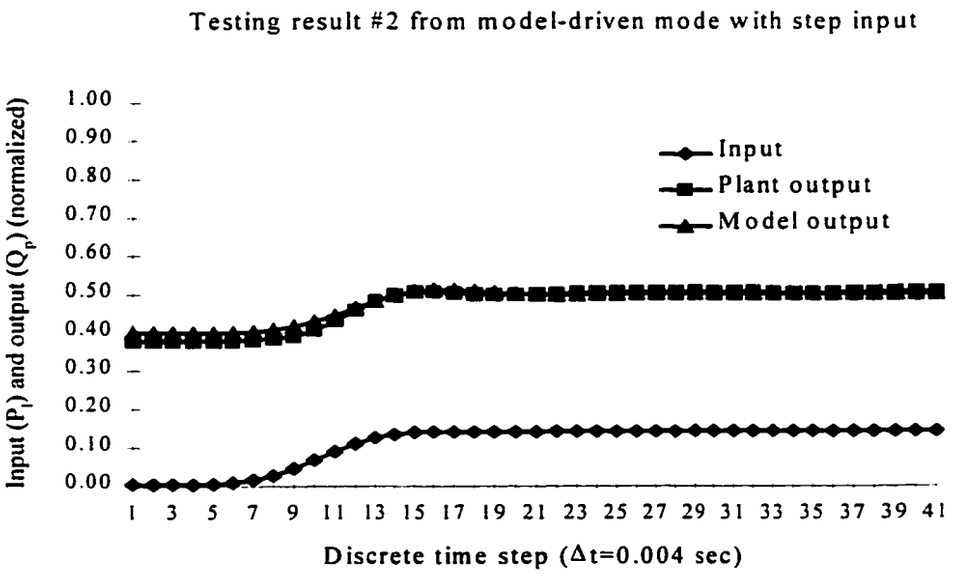


Figure 7.10(a) Model-driven testing result #2

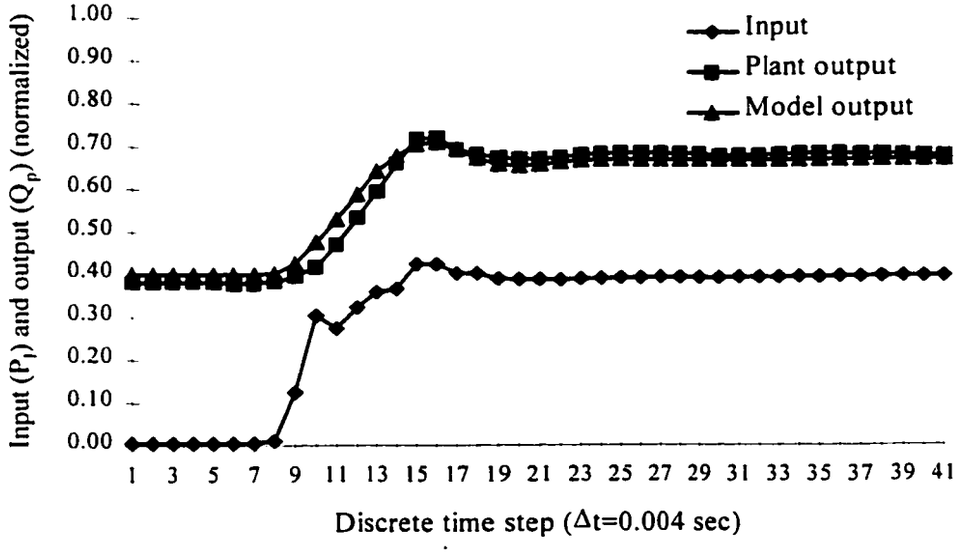


Figure 7.10(b) Model-driven testing result #6

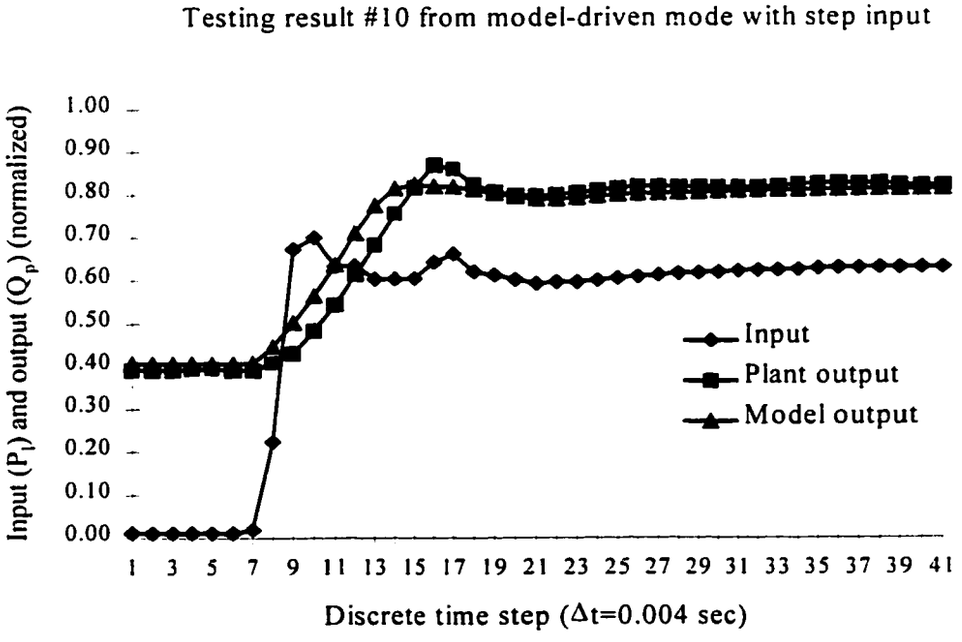


Figure 7.10(c) Model-driven testing result #10

Figures 7.10 Model-driven testing results using step inputs

7.5 Establishment of Two-Input One-Output Model of the Load Sensing Pump System

The morphology of the neural network for the second model of the load sensing pump system is shown in Figure 7.11, where the pump supply pressure, P_s, was used as an input to the model in addition to P_l. As discussed in Section 5.2, P_l and P_s were not independent signals in that P_s followed P_l via Q_p. The pump can be identified but only over a certain load operating condition. The same procedures with the same training and testing data for the first model were carried out to establish the second neural network model. The error function during the training monotonically reduced and finally converged to the order of 8.0×10^{-5}. All testing results are presented using NRMS tables and plots similar to the first model.

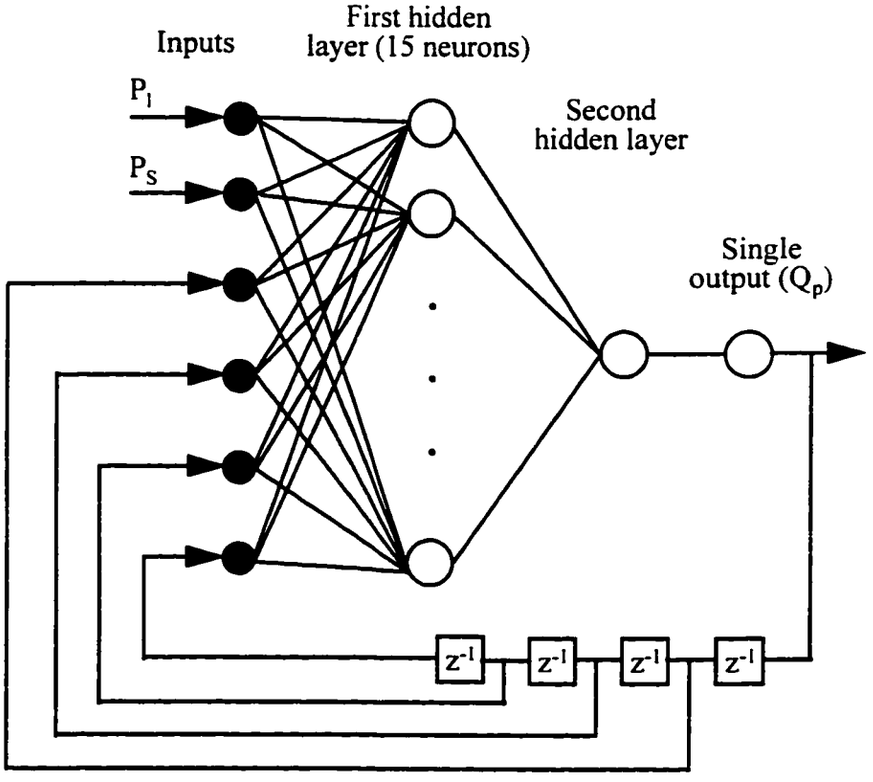


Figure 7.11 Morphology of two-input single-output neural network model

7.5.1 Model Validity Testing with the Random Input

Figures 7.12(a), (b) show the testing results from the plant-driven mode and Figures 7.13(a) and (b) for the model-driven mode. Again, only a portion of 200 data pairs are shown in each plot for clarity purpose. The corresponding NRMS values are summarized in Table 7.4.

Table 7.4 Second model testing results with random input

Input type	NRMS (plant-driven)	NRMS (model-driven)
50 Hz random	0.0148106	0.022329

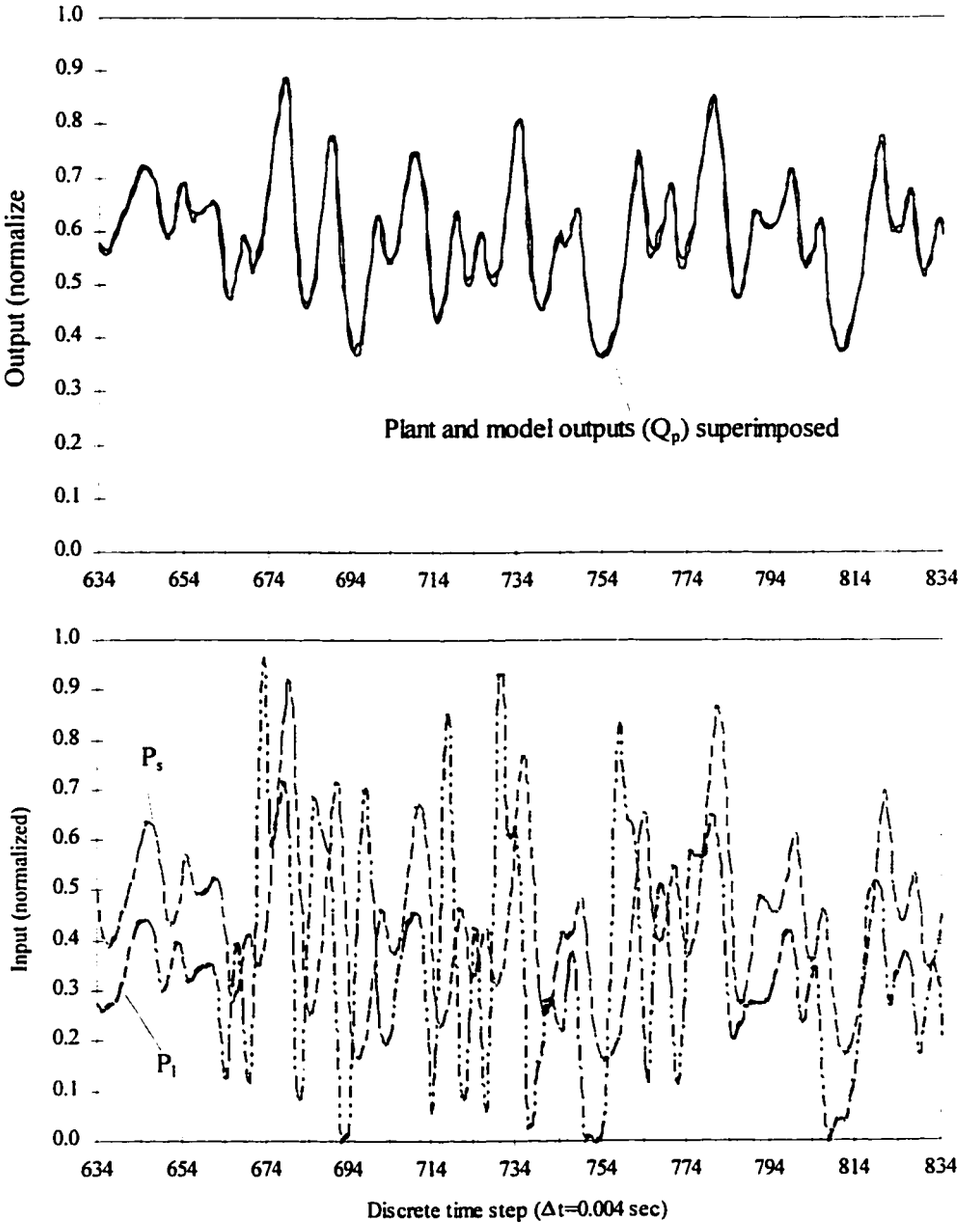


Figure 7.12(a) Testing result from plant-driven mode with random input (time step: 634-834)

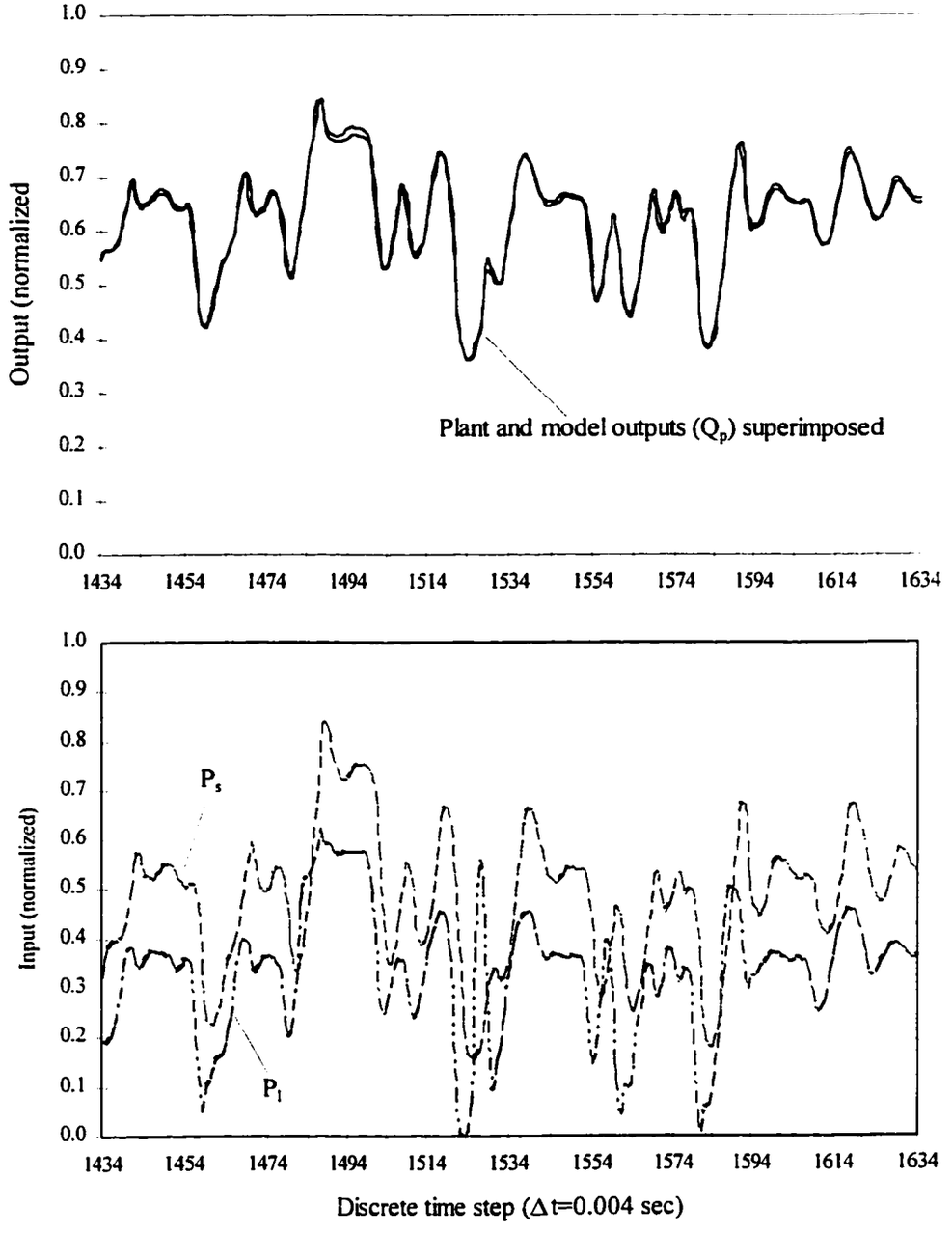


Figure 7.12(b) Testing result from plant-driven mode with random input (time step: 1434-1634)

Figure 7.12 Plant-driven testing results using random input (two-input single-output model)

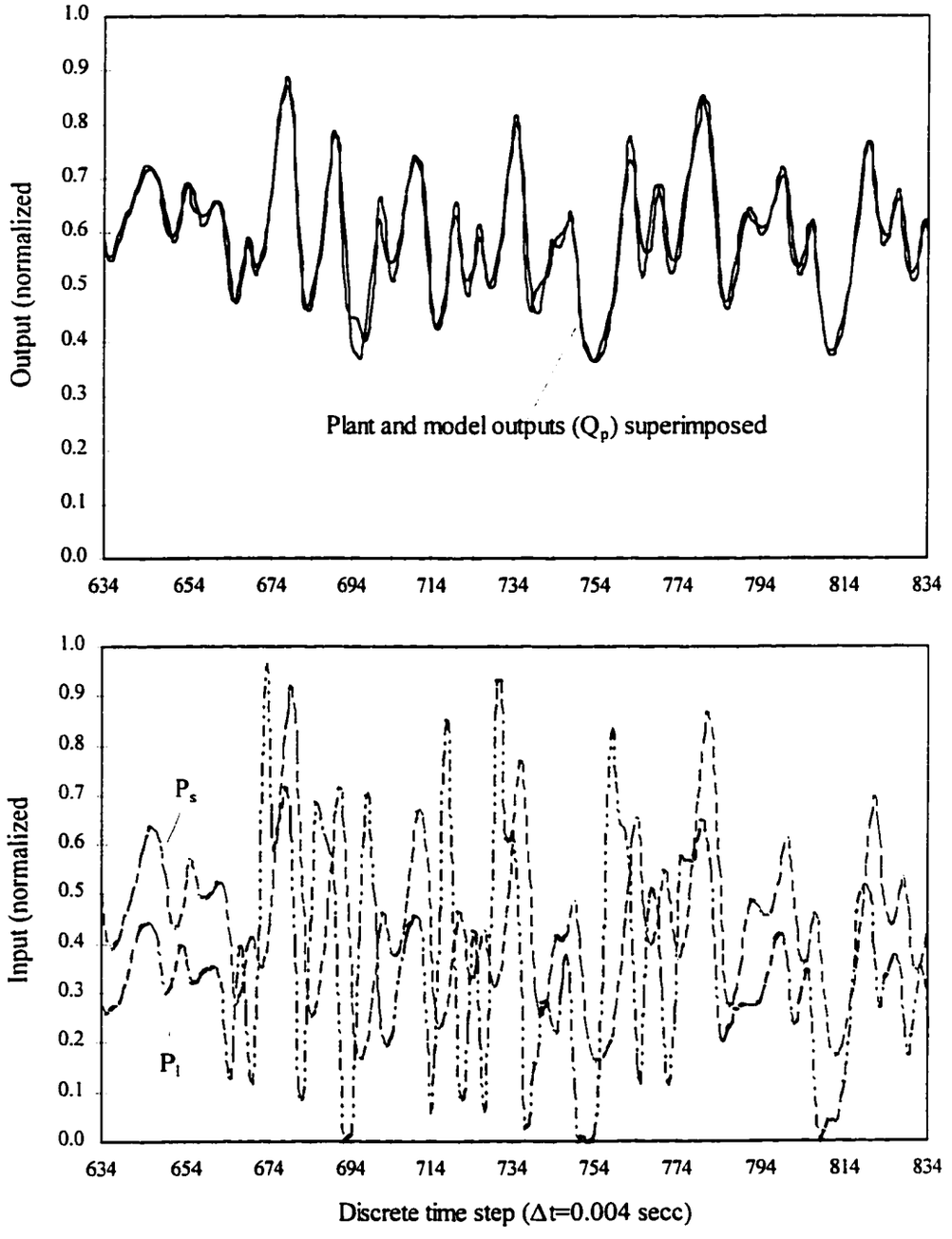


Figure 7.13(a) Testing result from model-driven mode with random input (time step: 634-834)

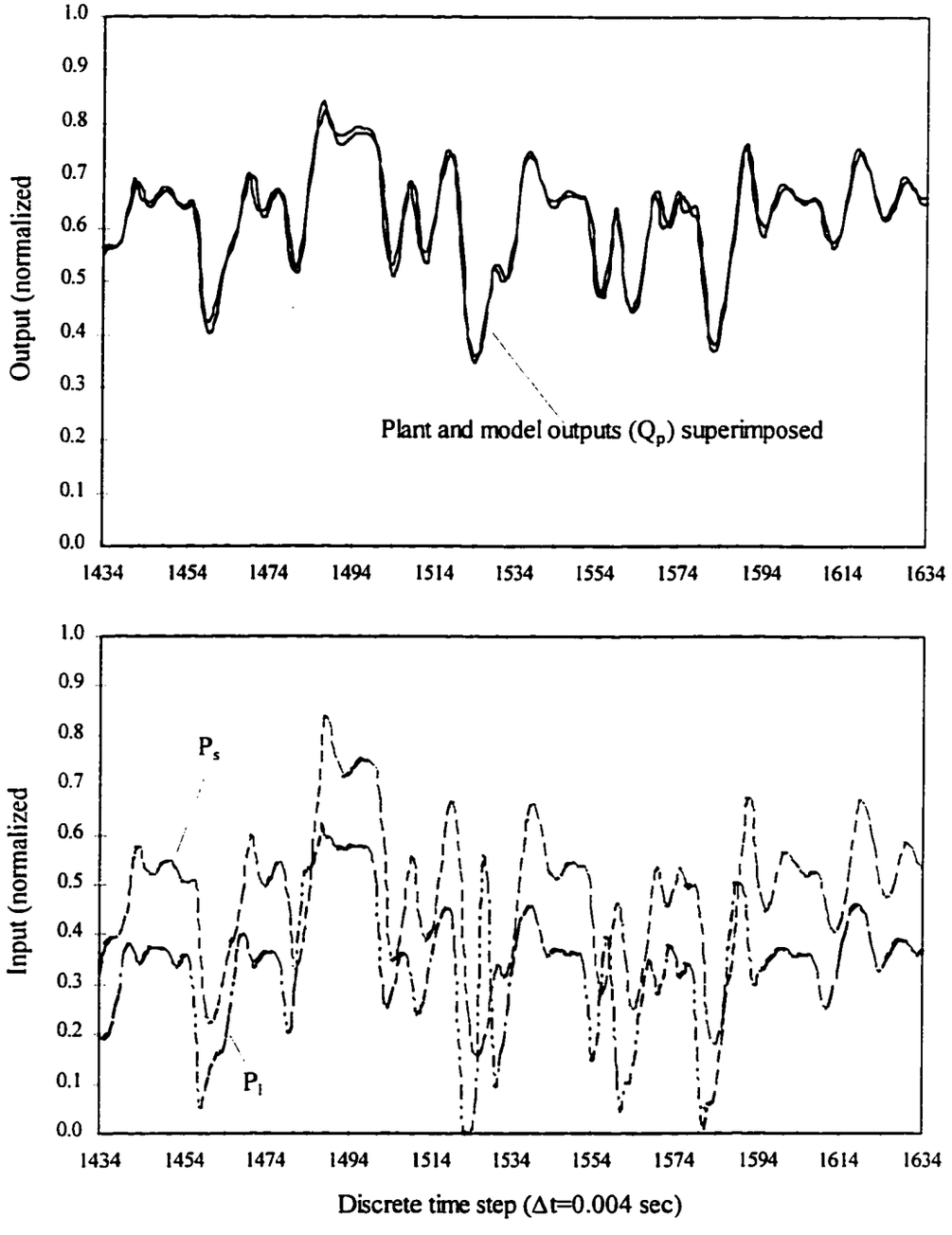


Figure 7.13(b) Testing result from model-driven mode with random input (time step: 1434-1634)

Figure 7.13 Model-driven testing results using random input (two-input single-output model)

7.5.2 Model Validity Testing with Swept Sine Wave Input

Second model testing results with swept sine wave inputs are summarized in Table 7.5. Only results from test#4 and test#8 are illustrated in Figures 7.14 and 7.15. The two plots in Figures 7.14 present the testing results from test#4 with the swept input from 0.001 to 1.0 Hz in the plant-driven mode (Figure 7.14(a)) and the model-driven mode (Figure 7.14(b)). The plots in Figures 7.15(a) and (b) show the testing results in plant-driven mode only for the swept inputs of 1-10 Hz and 25-40 Hz, respectively; the plots in Figures 7.15(c) and (d) are for the same swept inputs, but in the model-driven mode.

Table 7.5 Second model testing results with sweeping sinusoidal inputs

Input type: swept sine wave (test#1 - test#4: 0.001-1.0 Hz test#5 - test#8: 1.0 - 60 Hz)	NRMS (plant-driven)	NRMS (model-driven)
test#1	0.00506	0.00566
test#2	0.00220	0.00269
test#3	0.00205	0.00255
test#4	0.00295	0.00340
test#5	0.02346	0.02883
test#6	0.02086	0.02618
test#7	0.01704	0.02603
test#8	0.01000	0.01448

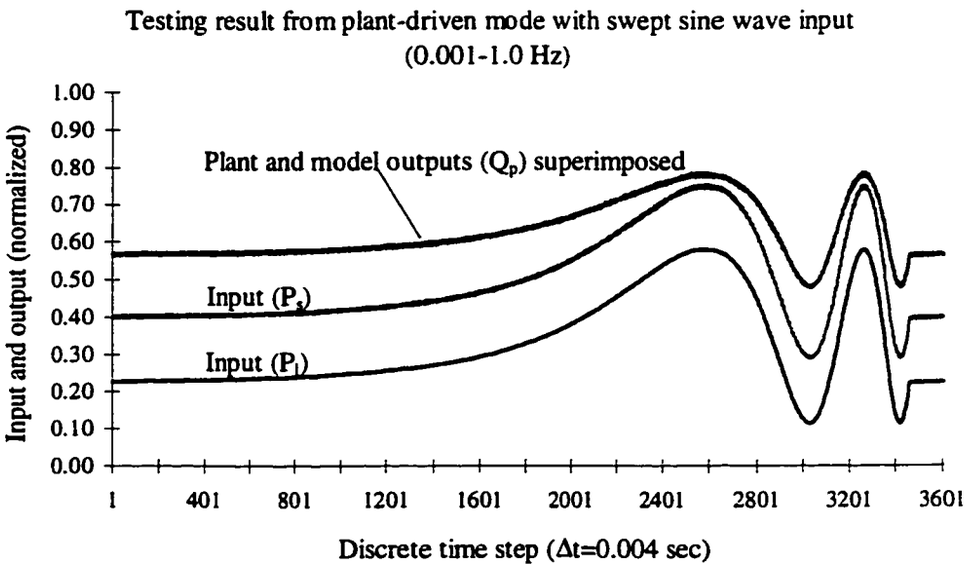


Figure 7.14(a) Plant-driven testing result using swept sine wave (0.001 - 1.0 Hz)

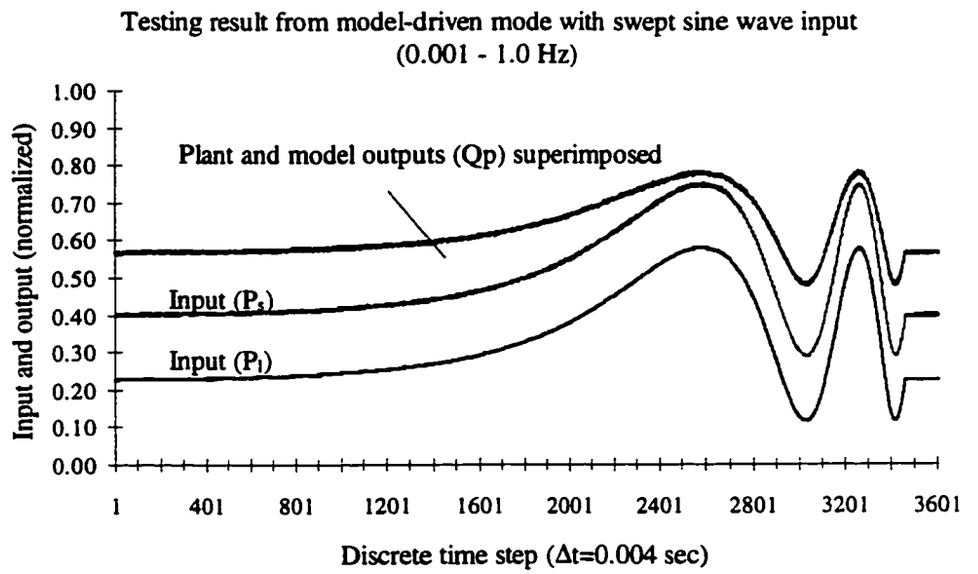


Figure 7.14(b) Model-driven testing result using swept sine wave (0.001 - 1.0 Hz)

Figure 7.14 Testing results using swept sine wave input (two-input single-output model)

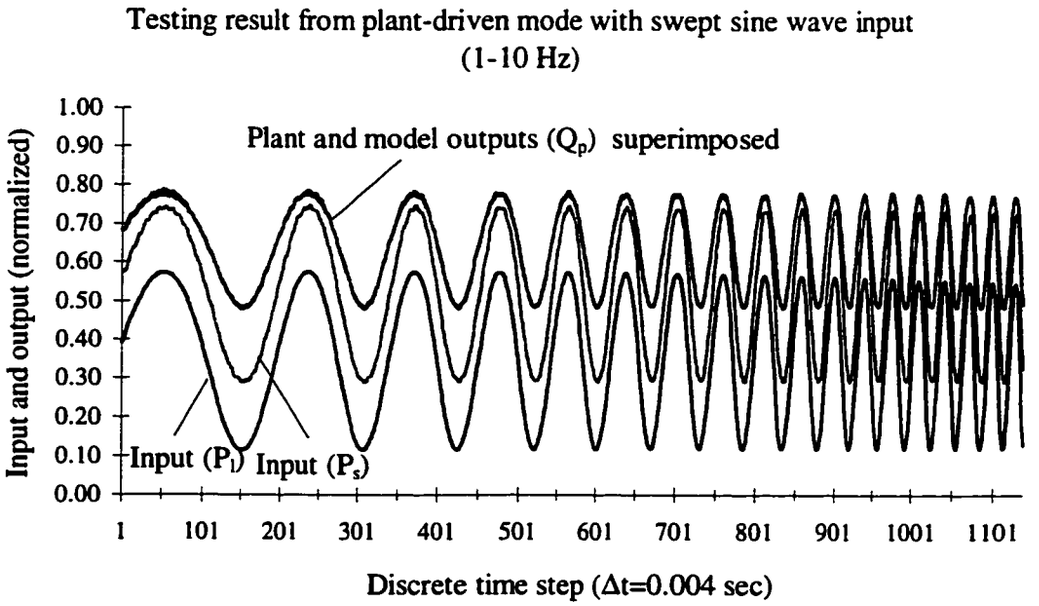


Figure 7.15(a) Plant-driven testing result (1 - 10 Hz)

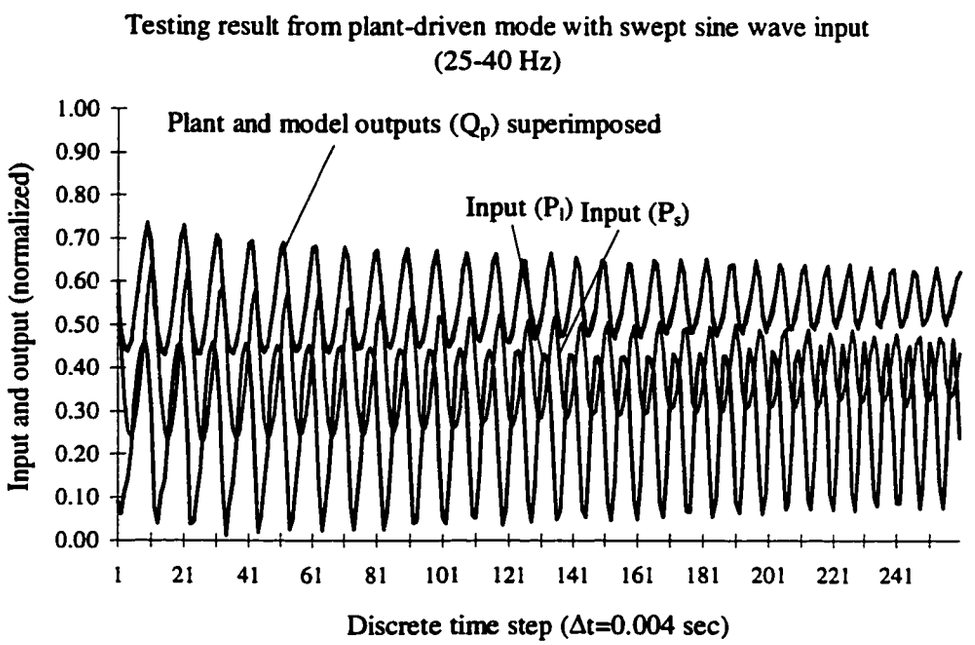


Figure 7.15 (b) Plant-driven testing result (25 - 40 Hz)

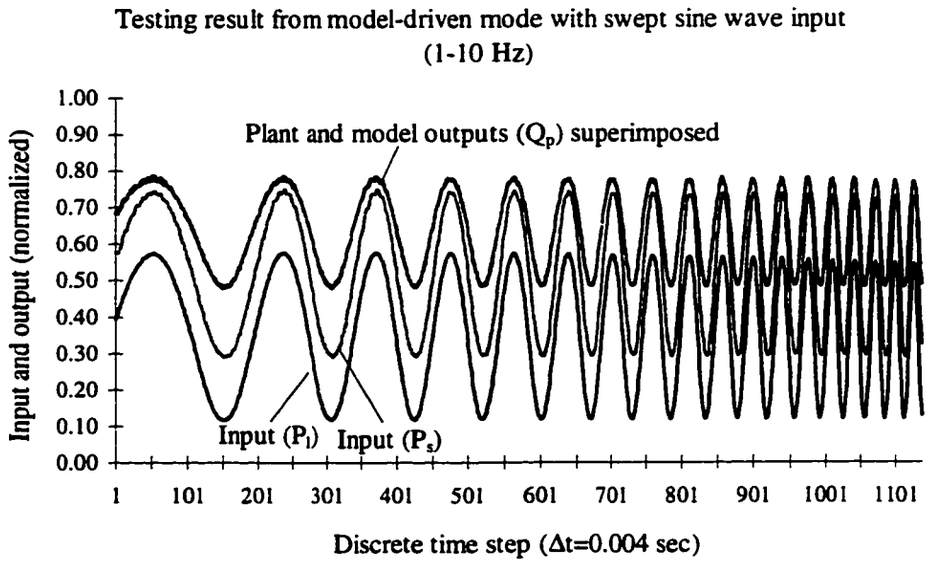


Figure 7.15 (c) Model-driven testing results (1-10 Hz)

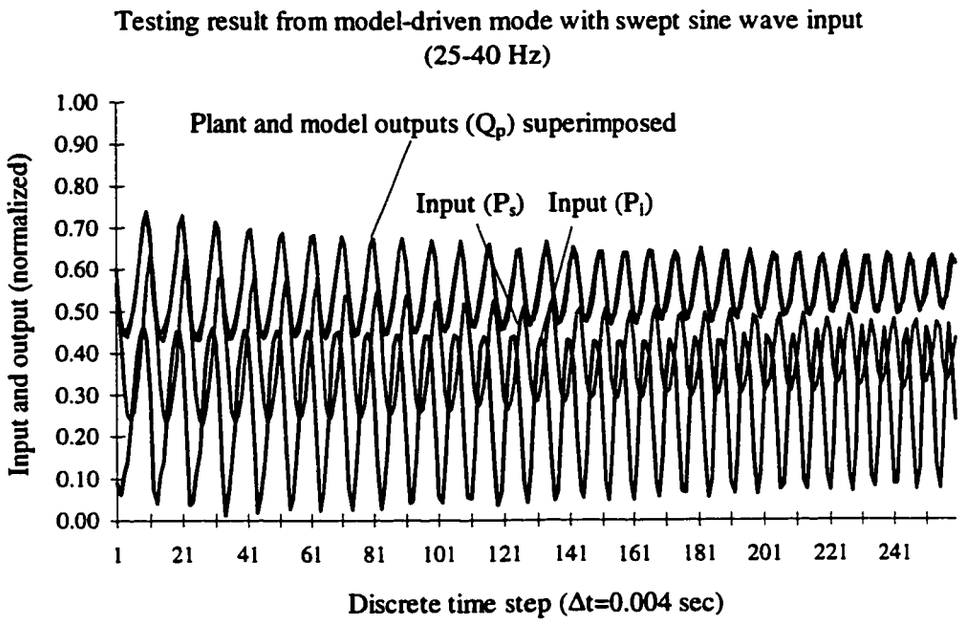


Figure 7.15(d) Model-driven testing result (25 - 40 Hz)

Figure 7.15 Testing results using swept sine wave input (two-input single-output model)

7.5.3 Model Validity Testing with Step Inputs

The second model testing results with step inputs are summarized in Table 7.6. Three plots in Figures 7.16(a) -(c) graphically illustrate the three typical testing results (test#2, test#6, test#10) in the plant-driven mode, and Figures 7.17(a) - (c) in the model-driven mode.

Table 7.6 Second model testing results with step inputs

Input type quasi-step input	Input magnitude P_I (steady state values)	NRMS (plant-driven)	NRMS (model-driven)
test#1	0.109	0.00858	0.01198
test#2	0.146	0.00806	0.01147
test#3	0.216	0.00751	0.01081
test#4	0.279	0.00828	0.01190
test#5	0.349	0.00712	0.00952
test#6	0.411	0.00808	0.00992
test#7	0.459	0.03203	0.03487
test#8	0.523	0.02110	0.02062
test#9	0.577	0.02010	0.01947
test#10	0.634	0.01631	0.01644

Comparing the training and testing results for the second model to that for the first model, it is shown that the second model can also be established equally successful. It is interesting to notice that the results from tests #8 - #10 showed an obvious improvement on the accuracy compared to the same tests earlier for the SISO model. Refer to Figure 7.10(c) and Figure 7.17(c) for example. Since the second model used the same training and testing data as for the first model, and had reached the same degree of the training error function, this improved accuracy for the second model must be a consequence of the additional input signal of pump pressure P_s. It may be that, because

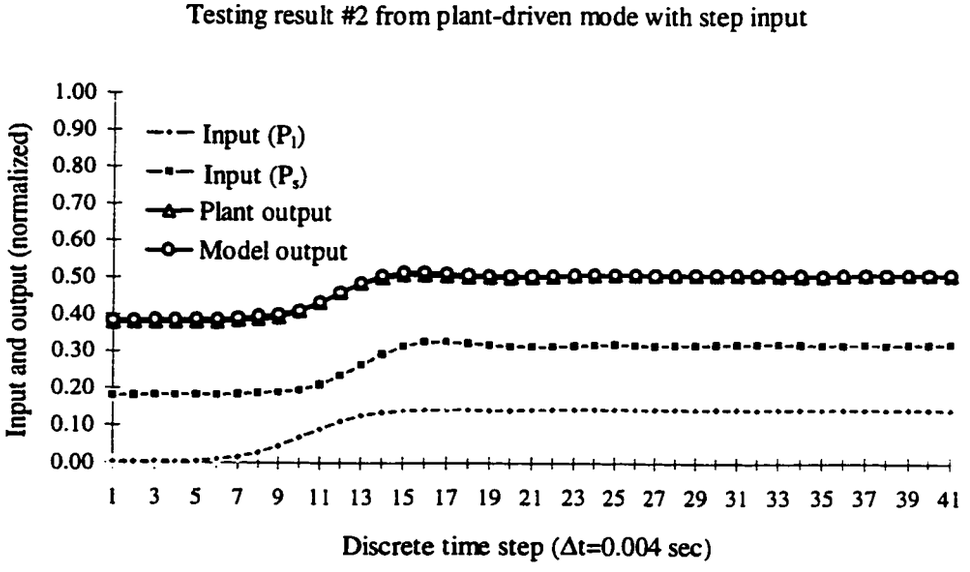


Figure 7.16(a) Plant-driven testing result #2

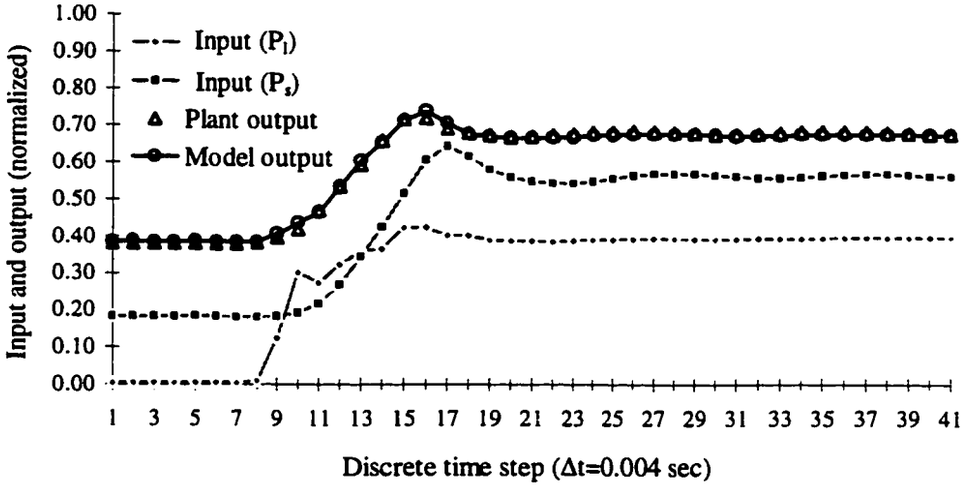


Figure 7.16(b) Plant-driven testing result #6

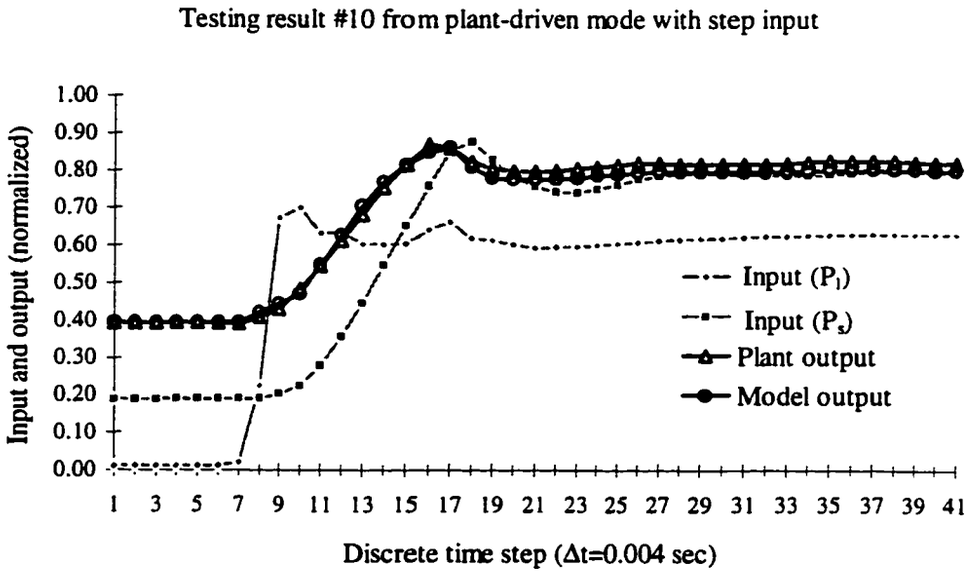


Figure 7.16 (c) Plant-driven testing result #10

Figures 7.16 Plant-driven testing results using step inputs

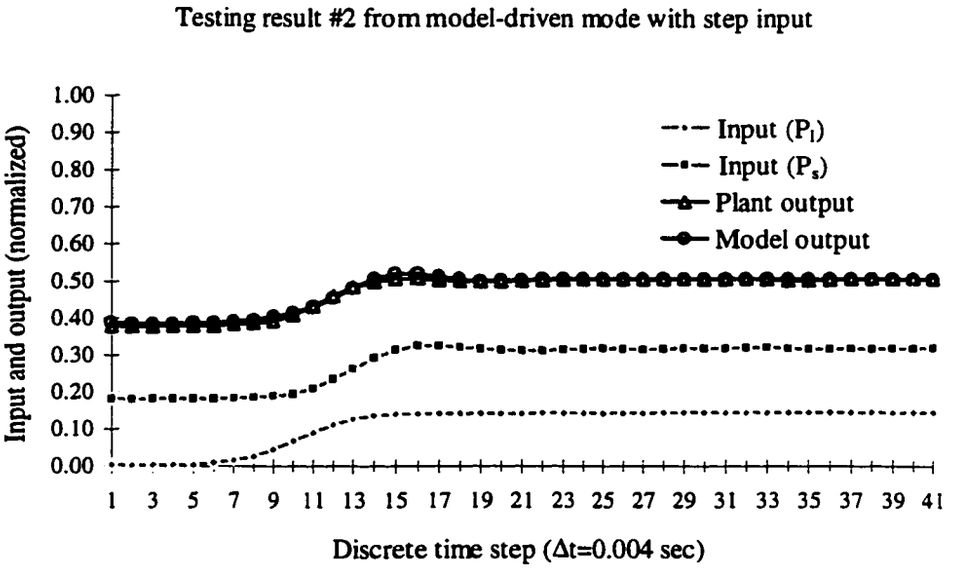


Figure 7.17 (a) Model-driven testing result #2

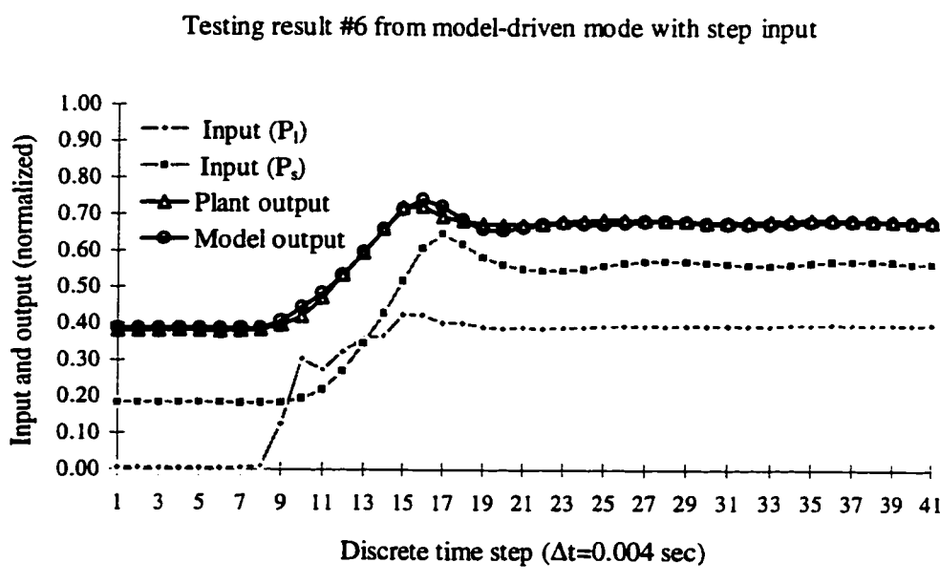


Figure 7.17 (b) Model-driven testing result #6

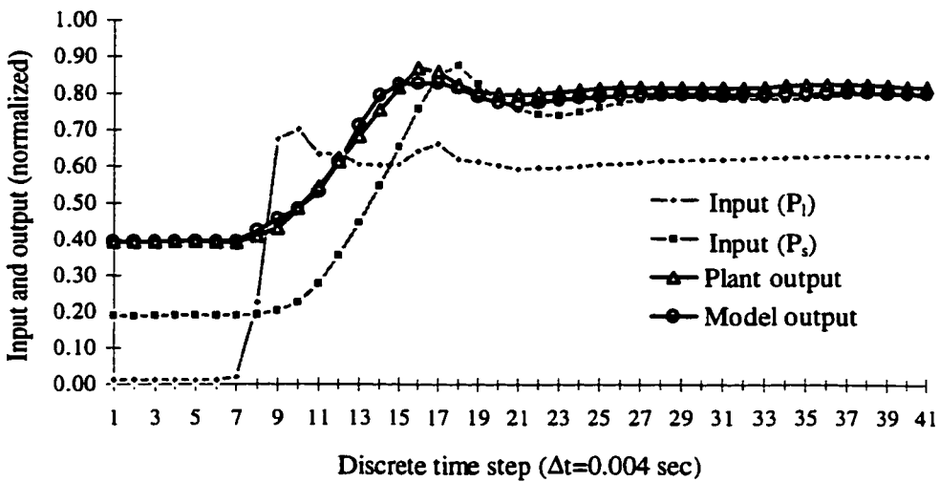


Figure 7.17 (c) Model-driven testing result #10

Figures 7.17 Model-driven testing results using step inputs

the number of true inputs to the second model was increased from 1 out of 5 (in the first model) to 2 out of 6 (note: in both models, 4 out of 6 were delayed model outputs "contaminated" by the modeling error), comparatively, the second neural model had more true inputs than that in the first model.

7.6 Discussion

Models of the load sensing pump system have been established in two different configurations using the experimental identification approach. The validation tests show that the two trained models are able to approximate the pump flow output in both the transient and steady states, because of their interpolating capabilities developed throughout the training or learning process. The "goodness" of the model should be measured by how accurately the model can approximate the plant outputs over the entire trained range. The modeling accuracy was found very satisfactory over most of the specified operating range, with a reduction in the larger step responses. A further discussion on the model quality is now necessary to analyze what has affected or has limited the modeling accuracy in practice. The following discussion will be focused on two aspects, training accuracy and error accumulation, as the two appear to be the most critical factors in analyzing and interpreting the obtained model accuracy.

7.6.1 Training Accuracy

As has been pointed out in the modeling error analysis in Chapter 5, the overall modeling accuracy for a given plant is dictated by the training accuracy and the sample rate. With a properly chosen sample rate, the modeling accuracy mainly depends on the training accuracy, which in turn depends on the data accuracy and the data distribution in both frequency and magnitude.

When using experimental data for training, noise embedded into the data inevitably imposes a physical limit on the achievable training accuracy. It is impossible

to train a neural model for an accuracy higher than the data accuracy. In other words, the best training accuracy can only be expected to be, ideally, of the same degree of the data accuracy. The estimated training accuracy in this study was based on the plant-driven testing result with the random inputs, as calculated to be the average of absolute error over 2000 data pairs. The estimated training accuracy for both models was within a range of 1.07 % -1.19 %. Comparing this value to the data accuracy of about 1.06% (calculated based on the sampling error, calibration error and the average fitting error), the training accuracy was comparable to the data accuracy. It can be therefore concluded that the neural models have been trained to their best accuracy on average, and more accurate training results would not be realistic. This also demonstrated the neural network's capability of "best fitting" noisy data in nonlinear approximations.

The poorer accuracy observed in the model-driven testing as shown in Figure 7.10(c) is associated with a slight decrease in the accuracy of the plant-driven testing shown in Figure 7.9(c). The accuracy decrease in the plant-driven testing can be attributed to the poorer training accuracy over that particular region (normalized step responses larger than 0.73). This problem is often caused by insufficient training data points over that local region. To further analyze this situation, consider the histograms of the training output (pump flow rate Q_p) and input (load pressure P_l) in their normalized values shown in Figures 7.18(a) and (b). These plots clearly show that actual signal magnitudes are non-uniformly distributed over their full ranges. In Figure 7.18(a), about 95% data points were clustered in the range less than 0.73. This means that although the training was performed in the desired full range of 0.4-0.9 as pre-specified in the normalized pump flow output, it had been unfairly concentrated on the range of 0.4-0.73. Consequently, the trained model showed good accuracy over this region, but poorer accuracy over the region where the normalized magnitude of Q_p was larger than 0.73.

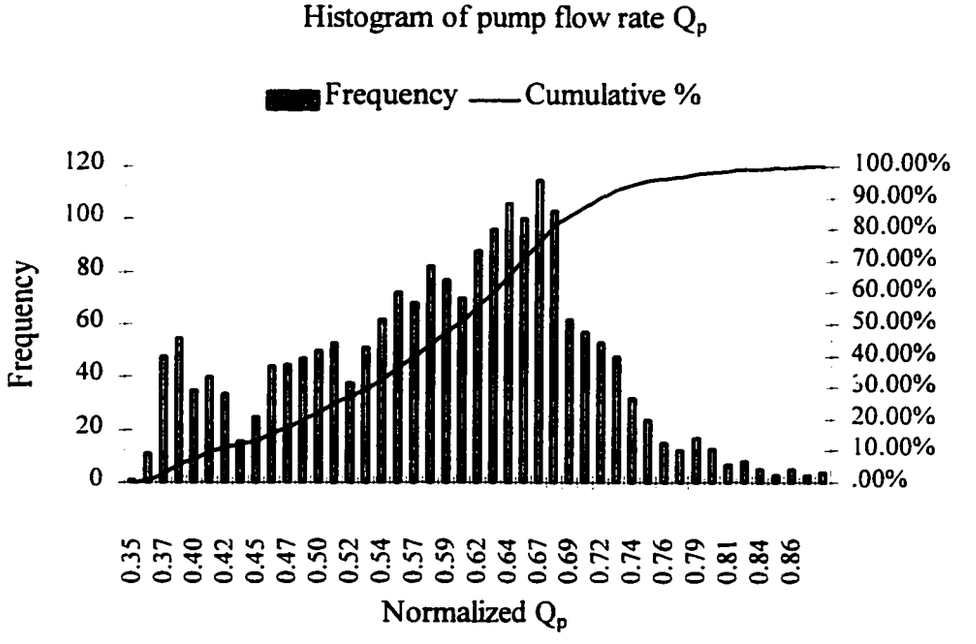


Figure 7.18(a) Histogram of pump flow rate Q_p

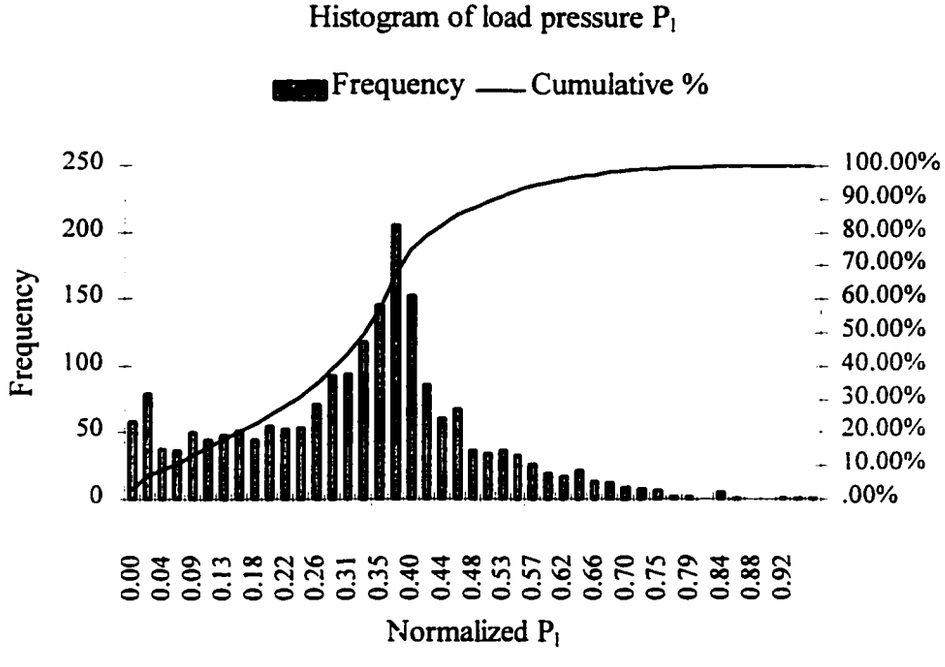


Figure 7.18(b) Histogram of load pressure P_l

Figure 7.18 Histograms of training data

The scarcity of the output signal over the region of 0.73-0.9 can be explained by examining the distribution of the input signal P_l on the histogram plot in Figure 7.18(b). As has been mentioned, the full range of the pump output was 0.4 -0.9 corresponding to the actual flow rate of 8 to 18 GPM in the experiment. To excite the pump system operating at the full range, the input pressure signal P_l was required to operate over 0.0-0.8 corresponding to a pressure of 0- 2000 psi with, ideally, a uniform distribution. The histogram of P_l, however, reveals that the actual load pressure distribution was non-uniform with the highest frequency at about 0.4. Although the normalized pressure values have been expanded to 0.95, more than 98 % of the points were within a range of 0-0.7. This means that most of the training had been carried out over the region 0.0-0.7 and highly concentrated at about 0.4 (1000 psi). It can be, therefore, understood that the scarcity of input data over large amplitude was the most logical reason for insufficient training, hence a poorer local modeling accuracy.

In addition, the non-uniform distribution of load pressure signal over the range of 0-0.9 was due to the effects of filtration and the dynamic characteristics of the pressure control valve. Clearly, the physical system has imposed a constraint in the experimental data distributions. One way to alleviate this practical problem is to further expand the pump system operating range (in magnitude) so that the concentrated training region can be also further expanded to cover the desired modeling range. But caution must be taken because of excessive transient pressures that could cause damage to the experimental system. This aspect will be considered in the future work to be discussed in the Chapter 8.

7.6.2 Error Accumulation

Error accumulation has been observed throughout all testing results presented above by comparing the plant-driven tests and model-driven tests. The error accumulation is considered to be a major cause for the local poorer modeling accuracy

observed in Figure 7.10(c). The degree of the accumulation was affected by the training accuracy obtained, and the sample rate which was associated with the number of accumulations over a time period. Since it is practically impossible to have a model structure that perfectly matches with the real plant structure, and to have perfect experimental data, there always exists an error between the model and plant output in the training. As long as this error exists, the error accumulation through the feedback paths in the tests will always magnify this error, resulting in poorer modeling accuracy than the training accuracy. A properly chosen sample rate can effectively avoid significant error growth, but can never eliminate this problem. It should be understood that error accumulation is inherent in model structures that use model output feedback to capture plant dynamic characteristics. In other words, the error accumulation is inevitable for the recurrent structure models, unless the sample rate is so slow that only steady state of the plant is modeled. However, as long as the model can be trained to have the error minimized to a sufficiently small value and the resulting error growth is, comparatively, insignificant, then the established model can often be accepted, and the error accumulation problem no longer needs to be concerned.

If the experimental data are considerably noisy, then the error accumulation may cause severe problems in the final accuracy. Under this situation, particularly when there is not much one can do to the experimental systems to further improve the data precision, which is true in some applications, it would be natural to require that the model, after well trained, be able to interpolate the real plant outputs without further decrease in the modeling accuracy. In other words, it is the best that the model accuracy is limited only by the actual data accuracy. This is possible only if the model output feedback is no longer used in the structure to eliminate the error accumulation problem. Then, the question is: can a dynamic system be experimentally identified with a discrete model structure that only uses sufficient number of delayed inputs to capture the system dynamics. This is considered as a part of future work to be discussed in the next Chapter.

7.7 Summary

The experimental results presented in this Chapter has successfully established the feasibility of modeling a physical nonlinear dynamic plant using experimental data and a partially recurrent neural network, and the effectiveness of conjugate gradient training algorithm when applied to experimental data . The key issue is the modeling accuracy. The most critical factor affecting the modeling accuracy is the experimental data quality: the precision and the distribution in both frequency and magnitude. It is the most critical that the experimental system be designed such that the hydraulic parameters (flow rate and pressure) can be well controlled in order to obtain high quality of data.

Chapter 8

Summary, Conclusions and Recommendations

8.1 Summary

The overall objective of research in this study was to develop a neural network simulation package to assist designers in the simulation and configuration of hydraulic circuits. The specific objective of this thesis was to explore the capabilities of neural networks to approximate the nonlinear dynamics of a load sensing hydraulic pump system using experimental approach. The established neural net model is to be used as an simulator of the system, which, consequently, imposes the most restrictive requirements on the model performance in its accuracy and generalization property over the entire modeling range. The nonlinear system identification technique and the conjugate gradient algorithm were used in the calculation of the neural model parameters (neural net weights). A partially recurrent neural network morphology was adopted, and the NNARX scheme was used in the training. A load sensing pump system was selected as the physical nonlinear system to be identified.

In this thesis, the neural network modeling problem was first formalized through the approximation approach, which was based on the premise that a physical plant can never be completely known and perfectly represented by a set of mathematical models with limited numbers of parameters; a model can only be pursued as a best approximation of a real plant in an engineering sense. This approach has naturally led to the criteria for the justification of the model quality: the closeness of model outputs to the plant outputs, as both are subjected to the same testing inputs. The closeness is measured by modeling error or modeling accuracy. The modeling error is composed of

two error sources: the experimental data noise and the structure error caused by the virtual differences between the model structure and the unknown plant structure.

A simulation study was then conducted using noise-free data generated from simulated nonlinear dynamic plants. The purpose of this part of this study was to investigate ultimately achievable modeling accuracy, as "perfect data" were used, and to observe the structure error behaviors. Following the simulation examples, a systematic analysis on the modeling errors (or precisely structure errors) was performed to reveal how the error was back propagated and accumulated through feedback paths. The significance of this study is that it first studies the structure errors (often overlooked in this area when the hypothesis that a true mathematical description of the plant exits is used), and provides a clear insight into the mechanism on how the modeling errors are formed when a recurrent model structure is used. This study established the feasibility of using a neural net model to emulate nonlinear system dynamics, and suggested a means to effectively reduce the error growth.

The established concepts and approach were subsequently applied to the identification of a hydraulic load sensing pump system. An identification experimental system was designed and constructed to test the dynamics of the load sensing pump. Particular attention was paid to the design and generation of sufficiently rich input signals. The experimental data were sampled at appropriate rate and fed into the neural models for the estimate of the model parameters. Since the modeling error observed was the combination of structure error and data noise, this part of the study examined the practically achievable modeling accuracy from an experimental point of view. The trained models were extensively tested and the results showed that the estimated models were able to emulate nonlinear dynamic responses of the pump with satisfactory accuracy over the well trained range. The training error and the error accumulation were the two most critical aspects in examining and interpreting the overall modeling accuracy. The experimental implementation also revealed some practical constraints

imposed by the available test facilities, which are to be resolved in the future work as recommended in the following sections.

In summary, the work presented in this thesis has investigated the use of neural networks in the identification of a particular nonlinear dynamic system for emulation purposes. It has been established through simulation and experimental studies that a partially recurrent neural net model, after successfully trained, is capable of approximating the dynamic behavior of a hydraulic load sensing pump with very satisfactory accuracy. The observed constraints on the experimental modeling accuracy are mainly from the availability of test devices.

8.2 Conclusions and Contributions

As briefly mentioned, the specific objective of this thesis was to explore the inherent capabilities of a neural network to approximate dynamic characteristics of a nonlinear system by developing a dynamic neural model for a nonlinear hydraulic component using experimentally measured input-output data. Based on the results of this study, the following specific conclusion is drawn:

A partially recurrent neural network is a suitable model structure for the identification of the nonlinear dynamics of the load sensing pump system. Central to this structure is a multilayer feed forward neural network that has been proven to be a "universal approximator" of nonlinear functions. Addition of input and output time delay lines provides the model with the dynamic features. Another advantage of this structure is the parsimony.

As a consequence of implementing the neural network modeling, other pertinent conclusions can be drawn:

(1) The training algorithm developed from the error backpropagation and conjugate gradient method is shown, in conjunction with NNARX scheme, to provide fast convergence and consistent stable solutions, as has been observed in the preliminary training experiments.

(2) For the recurrent types of model structures, the overall modeling accuracy depends on the training accuracy and is also affected by the error accumulation through the recurrent paths. The training accuracy in turn depends on the experimental data quality (i.e. data precision, uniform distribution, and richness in frequency and magnitude). Proper sampling rate can reduce the effect of the error accumulation, but cannot eliminate this effect because it is inherent to the recurrent model structures. If the overall modeling accuracy is acceptable, then the error accumulation is not of concern. Otherwise, a different model structure other than recurrent types has to be considered.

Certain relevant and significant observations were noted:

(1) It was found that the filtration of high frequency components in the experimental data can help in getting rid of data noise that do nothing but 'confuse' the neural models in the training process. At the same time, however, the filtration changes the magnitude distribution of the input signals to a non-uniform type which in this study caused the training to be intensified over the concentrated data region.

(2) Given the load sensing pump system and an adequately chosen sample rate, the quality of experimental data is the key to the success in the modeling. Special care must be placed on the design and construction of the experimental system to ensure that the hydraulic parameters are under control.

The major contributions from this thesis work are listed below:

(1) The first major contribution was made through a systematic analysis on the modeling error accumulation problem associated with recurrent types of model structures. The problem of excessive modeling error in the transient response was first isolated. A theoretical analysis was then conducted to study modeling error behavior. This study successfully revealed the mechanism of how the error associated with each model output was backpropagated and accumulated through feedback paths of recurrent models, and then embedded into the next model output. The significance of this study was that it first identified clearly the modeling error accumulation problem to be a major cause for the deterioration of the modeling accuracy, and suggested a means to effectively reduce the error growth in order to improve modeling accuracy. The modeling error analysis contributed to the theoretical study in that it provided theoretical fundamentals to the research approach to this particular modeling problem.

(2) The second major contribution was made on the experimental implementation of the neural network approach to the modeling of a particular hydraulic component, a hydraulic load sensing pump, based on experimental measurements. The significance of this implementation was that the applicability of the neural network approach to modeling a "real-world" hydraulic component using actual data was investigated and that practical constraints imposed by the actual hydraulic experimental testing facilities (not revealed by theoretical or simulation studies) were studied. The experimental implementation successfully established, from a practical point of view, the feasibility of the neural network approach to modeling a real hydraulic component, and suggested that the hydraulic data quality in terms of data precision and data distribution could significantly affect final model accuracy.

In addition, although the richness of input signals was discussed extensively in published literature, the discussions were mainly focused on the frequency distribution of the input. For the experimental modeling of nonlinear dynamic systems over the entire operating range, the amplitude of the input should also be specified properly to ensure that it covers all possible magnitudes of inputs when the model is in practical use.

In this thesis, the richness of the input signals was clearly defined to be measured by the both of its frequency bandwidth and amplitude distribution over the modeling range. It was implemented experimentally in the generation and pre-processing of input signals to improve the 'richness', which has been shown to be very effective in establishing the neural network model capabilities of learning and generalization.

8.3 Recommendations for the Future Work

The use of neural networks to emulate nonlinear dynamics of a load sensing pump system has been successfully established through an experimental identification approach. However there is still modeling accuracy degradation observed locally with larger input magnitude, because of insufficient training data over that local region. It was found that the sampled data had a higher density at the center of the training range, which resulted in better accuracy of the model over the center part. This non-uniform data distribution is a consequence of the filtration from the physical devices in the experimental system. In order to improve the modeling accuracy with larger input signals, the first recommendation would be to further expand the testing system operating range (in magnitude) so that the concentrated training region can be also further expanded to cover the desired modeling range. This requires that the experimental system be modified to accommodate higher transient pressure and flow rates.

The error accumulation associated with recurrent model structures has been observed throughout the simulation studies and experimental modeling results. For very noisy data or higher accuracy requirements, it would be desirable to search for non-recurrent model structures that can essentially eliminate the error accumulation problem. One candidate of such model structure could be a nonlinear FIR (Finite Impulse Response) structure which has received particular attention from others in the area of identification with neural networks [Samad, T., 1996]. The second recommendation would be to investigate the use of the neural networks with FIR structure to model the

load sensing pump and other real dynamic systems as an emulator. With FIR structures, the neural networks have only input delays for dynamic features, and the current model response depends only on the current and previous inputs. Often, a large number of input delays have to be considered because the series of the input and its delays must be sufficiently long so that the truncation error would be minimized (or the inputs prior to the last delayed input would have negligible effects on the current response).

The interaction between the load sensing pump and other in-line hydraulic devices has prevented the load sensing pump itself from being identified over the entire modeling range, because of the dependency of pump supply pressure and load pressure. The third recommendation would be to modify the primary system such that the supply pressure can be actively controlled independent of the load sensing pump flow. Two un-correlated pressures can then be measured and used as inputs to the single load sensing pump dynamic model.

As a final recommendation, it is suggested that a study be initiated to implement and interface a neural net based model with a traditional model to simulate a complete system. Many problems associated with time steps must be overcome in the development of this hybrid type of simulation package.

References:

Astrom, K. J. and Eykhoff, P. (1971). System Identification — A Survey. *Automatica*, Vol.7, Pp. 123-162.

Battiti, R. and Masulli, F. (1990). BFGS Optimization for Faster and Automated Supervised Learning. *INCC 90 Paris, International Neural Network Conference*, 2, Pp. 757-760.

Bellman, R. and Astrom, K. J. (1970). On Structural Identifiability. *Mathematical Biosciences* 7, Pp. 329-340.

Billings, S. A., Jamaluddin, H. B. and Chen, S. (1992). Properties of Neural Networks with Applications to Modeling Non-linear Dynamical Systems. *Int. J. Control*, Vol. 55, No. 1, Pp. 193-224.

Box, M. J. (1965). A New Method of Constrained Optimization and Comparison with Other Methods. *Computer Journal*, Vol. 8.

Chen, C. H. and Lai, H. (1993). A Comparison Study of the Gradient Descent and the Conjugate Gradient Backpropagation Neural Networks. *World Congress on Neural Networks*, Oregon Convention Center, Portland, Oregon, Pp. III-401.

Chen, S. and Billings, S. A. (1992). Neural Networks for Dynamical System Modeling and Identification. *Int. J. Control,* Vol. 56, No. 2, Pp. 319-346.

Chen, S. and Cowan, C. F. N., Billings, S. A., Grant, P. M. (1990). Parallel Recursive Error Algorithm for Training Layered Neural Networks. *Int. J. Control*, Vol. 51, No.6, Pp.1215-1228.

Chen, S., Billings, S. A. and Grant, P. M. (1990). Non-linear System Identification Using Neural Networks. *Int. J. Control*, Vol. 51, No. 6, Pp. 1191-1214.

Cybenko, G. (1989). Approximation by Superpositions of a Sigmoidal Function. *Mathematics of Control, Signals, and Systems*, 2(4), Pp. 303-314.

Dracopoulos, D. C. and Jones, A. J. (1993). Modeling Dynamic Systems. *World Congress on Neural Networks*, Oregon Convention Center, Portland, Oregon, Pp. III-289.

Farlow, J. S. (1984). *Self-organized Methods in Modeling —GMDH Type Algorithms.* Marcel Dekker, New York.

Godfrey, K. (1983). *Compartmental Models and Their Applications.* Academic Press.

Grewal, M. S. and Glover, K. (1976). Indentifiability of Linear and Nonlinear Dynamical Systems. *IEEE Transactions on Automatic Control*, December, Vol. AC-21, Pp.833-837.

Gustavsson, I., Ljung, L. and Soderstrom, T. (1977). Survey Paper: Identification of Process in Closed Loop — Indentifiability and Accuracy Aspects. *Automatica*, Vol. 13, Pp. 59-75.

Haber, R. and Unbehauen, H. (1989). Structure Identification of Nonlinear Dynamic Systems — A Survey on Input/Output Approaches. *Automatica*, Vol. 26, No. 4, Pp. 651-677.

Hadaegh, F. Y. and Bekey, G. A. (1985). Near-Identifiability of Dynamic Systems. *Mathematical Biosciences* 77, Pp. 325-340.

Hopsan User's Manual (1985). University of Linkoping. LITH - IKP _ R - 388, Revised.

Hornik, K. (1991). Approximation of Capabilities of Multilayer Feedforward Networks. *Neural Networks*, Vol. 4, Pp.251-257.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer Feed Forward Networks are Universal Approximators. *Neural Networks*, 2(5), Pp. 359-366.

Hsia, T. C. (1977). *System identification- Least-Squares Methods.* Lexington Books.

Hush, D. R. and Horne, B. G. (1993). Progress in Supervised Neural Networks - What's New Since Lippmann ? *IEEE Signal Processing Magazine*, January, Pp. 8-39.

Jin, L., Gupta, M. M. and Nikiforuk, P. N. (1996). Chapter 10, Approximation Capabilities of Feedforward and Recurrent Neural Networks. *Intelligent Control Systems - Theory and Applications*, Edited by Gupta, M. M. and Sinha, N. K., IEEE Press, New York, Pp.234-264.

Jin, L., Nikiforuk, P. N. and Gupta, M. M. (1994). Adaptive Control of Discrete — Time Nonlinear Systems Using Recurrent Neural Networks. *IEEE Proceedings-D: Control Theory and Applications*, Vol. 141, No. 3 May, Pp. 169-176.

Johansson, E. M., Dowla, F. U., and Goodman, D. M. (1992). Backpropagation Learning for Multilayer Feedforward Neural Networks Using the Conjugate Gradient Method. *International Journal of Neural Systems*, Vol. 2, No. 4, Pp. 291-301.

Levin, A. U. (1993). Predicting with Feedforward Neural Networks. *World Congress on Neural Networks*, Oregon Convention Center, Portland, Oregon, Pp. III-385.

Ljung, L. (1987). *System Identification - Theory for the User*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632.

Luenberger, D. G. (1984). *Linear and Nonlinear Programming, Part II.* Second Edition, Addsion-Wesley Publishing Company, London.

Møller, M. F. (1993). A Scaled Conjugate Gradient Algorithm for Fast Supervised Learning. *Neural Networks*, Vol. 6, Pp.525-533.

Mollo, J. R. (1990). Load-Sensing Pumps Have Their Time Come? *Hydraulic & Pneumatics Magazine*, May, 1990, Pp. 57.

Narendra, K. S. (1990). Adaptive Control Using Neural Networks. *Neural Networks for Control*, Edited by Thomas Miller, III, W., Sutton, R. S. and Werbos, P. J., The MIT Press, Cambridge, Massachusetts, London, England, Pp. 115-142.

Narendra, K. S., and Parthasarathy, K. (1989). Neural Network and Dynamical Systems -- Part II: Identification. Report No. 8902, Yale University.

Narendra, K. S., and Parthasarathy, K. (1990). Identification and Control of Dynamical Systems Using Neural Networks. *IEEE Trans. Neural Networks*, 1:4-27.

Nguyen, V. V. and Wood, E. F. (1982). Review and Unification of Linear Identifiability Concepts. *SIAM Review*, Vol. 24, No. 1, January, Pp. 34-51.

Pike, R. W. (1986). *Optimization for Engineering Systems*. Van Nostrand Reinhold Company, New York.

Powell, M. J. D. (1977). Restart Procedures for the Conjugate Gradient Method. *Mathematical Programming* 12, Pp. 241-254.

Reklatis, G. V., Ravindran, A., Ragsdell, K. M. (1983). *Engineering Optimization. Methods and Applications.* A Wiley-Interscience Publication, John Wiley and Sons. New York.

Richards, C.R., Tomlinson, S.P., Tilley, D.G., and Burrows, C.R. (1989). Bath fp - A Second Generation Simulation Package for Fluid Power Systems. *9th International Symposium on Fluid Power*, Cambridge, U.K.

Samad, T. (1996). Chapter 11, Neurocontrol - Concepts and Practical Considerations. *Intelligent Control Systems -Theory and Applications*, edited by Gupta, M. M., and Sinha, N. K., IEEE Press, The Institute of Electrical and Electronics, Inc. New York.

Sjoberg, Jonas (1993). *Regularization Issues in Neural Network Models of Dynamical Systems.* Ph. D. Thesis, Department of Electrical Engineering, Linköping University, Linköping, Sweden.

Soderstrom, T. and Stoica, P. (1989). *System Identification.* Prentice Hall, New York.

Tyson, F. C. (1961). *Industrial Instrumentation.* Prentice-Hall, Inc., Englewood Cliffs, NewJersey.

Widrow, B., and Lear, M. A. (1990). 30 Years of Adaptive Neural Networks Perception, Madaline and Backpropagation. *Proceedings of the IEEE*, Vol. 8, No. 9, September.

Xu, P., Ramden, T., Burton, R., Krus, P., and Sargent, C. (1994). Feasibility of Training a Neural Network Based on Hydraulic Component Simulator Using the Complex Method. *The 1994 International Fluid Power Workshop*, University of Bath, England.

Xu, X. P., Sargent, C. M., and Burton, R. T.(1996). Experimental Identification of a Flow Orifice Using a Neural Network and the Conjugate Gradient Method. *Transations of ASME: Journal of Dynamic Systems, Measurements, and Control,* June, Vol. 118, Pp. 273-277.

Zurada, J. M. (1992). *Introduction to Artificial Neural Systems.* West Publishing Company, New York.

Appendix A

Pressure Transducer Calibration Record

The objective of this appendix is to provide a record of the pressure transducers and their calibration results.

Model: Schaevitz, type P1021 -005

Serial: 113128

Range: 3500 psi.

Pressure limit: ≤5 x full range pressure or 12000 psi or whichever is less.

Burst Pressure: > 20 x full range pressure or 20000 psi whichever is less.

Mechanical natural frequency: 5 KHz min.

Date: Nov. 2, 1995

Table A.1 Pressure transducer calibration data record

P (psi)	Volts	P(psi)	Volts	P(psi)	Volts	P(psi)	Volts
0	0	1550	5.18	0	-0.02	1550	5.19
50	0.16	1600	5.35	50	0.16	1600	5.36
100	0.33	1650	5.52	100	0.33	1650	5.52
150	0.5	1700	5.68	150	0.5	1700	5.69
200	0.67	1750	5.85	200	0.67	1750	5.86
250	0.83	1800	6.02	250	0.84	1800	6.03
300	1.00	1850	6.18	300	1.00	1850	6.19
350	1.17	1900	6.35	350	1.17	1900	6.36
400	1.34	1950	6.52	400	1.34	1950	6.53
450	1.50	2000	6.68	450	1.51	2000	6.69
500	1.67	2050	6.85	500	1.68	2050	6.86
550	1.84	2100	7.02	550	1.84	2100	7.03
600	2.01	2150	7.18	600	2.01	2150	7.19
650	2.18	2200	7.35	650	2.18	2200	7.36

(To be continued)

Table A.1 Pressure transducer calibration data record (continued)

P (psi)	Volts	P(psi)	Volts	P(psi)	Volts	P(psi)	Volts
700	2.34	2250	7.52	700	2.35	2250	7.53
750	2.51	2300	7.68	750	2.516	2300	7.69
800	2.68	2350	7.85	800	2.68	2350	7.86
850	2.84	2400	8.01	850	2.851	2400	8.03
900	3.01	2450	8.18	900	3.019	2450	8.19
950	3.18	2500	8.34	950	3.18	2500	8.36
1000	3.34	2550	8.51	1000	3.35	2550	8.52
1050	3.51	2600	8.68	1050	3.52	2600	8.69
1100	3.68	2650	8.84	1100	3.68	2650	8.86
1150	3.84	2700	9.01	1150	3.85	2700	9.02
1200	4.01	2750	9.17	1200	4.02	2750	9.19
1250	4.18	2800	9.33	1250	4.19	2800	9.36
1300	4.35	2850	9.5	1300	4.35	2850	9.52
1350	4.51	2900	9.67	1350	4.52	2900	9.68
1400	4.68	2950	9.83	1400	4.69	2950	9.85
1450	4.85	3000	10.00	1450	4.85	3000	10.02
1500	5.02			1500	5.02		

Appendix B

Drag Flow Transducer Calibration Record

The objective of this appendix is to provide a record of the flow transducers and their calibration results.

Model: V-.5-AOS5K6-E.

Serial No.: 8710157, 8710158.

Range: 10 GPM (37.854 l/min).

Date: Jan. 27, 1996.

(Reading accuracy: ± 0.02 volts)

Table B.1 Drag flow transducer calibration data record

Flow (l/min) (Record[1])	Convert to (GPM)	Volts. (Record) 8710157	8710158	Temp (°C)
38.256	10.106	10.19	10.19	65.2
37.884	10.008	10.00	10.00	65.0
36.178	9.557	9.13	9.10	65.4
33.982	8.977	8.12	8.07	64.2
31.644	8.359	7.07	7.00	64.4
29.613	7.823	6.20	6.14	64.7
28.403	7.503	5.72	5.66	64.1
27.393	7.237	5.33	5.27	64.1
26.307	6.950	4.94	4.89	63.8
24.464	6.463	4.27	4.22	64.0
22.835	6.032	3.75	3.71	63.8
21.229	5.608	3.25	3.21	63.8
19.542	5.162	2.77	2.72	64.0

(To be continued)

[1] The data tabulated was rounded to three decimals. The actual data was recorded to five decimals through a computer operated calibration system.

Table B.1 Drag flow transducer calibration data record (continued)

Flow (l/min) (Record[2])	Convert to (GPM)	Volts. (Record) 8710157	8710158	Temp (°C)
17.173	4.537	2.15	2.11	63.4
15.690	4.145	1.82	1.77	63.0
14.334	3.787	1.52	1.47	63.0
12.609	3.331	1.19	1.14	62.8
11.697	3.090	1.02	0.98	62.4
10.766	2.844	0.86	0.82	62.4
9.590	2.533	0.69	0.66	61.9
8.575	2.265	0.56	0.54	61.5
7.035	1.859	0.38	0.37	60.9
5.859	1.548	0.27	0.26	60.6
4.541	1.200	0.17	0.17	59.7
3.872	1.023	0.134	0.131	58.8
3.472	0.917	0.116	0.114	58.3
1.615	0.427	0.036	0.038	55.7
0.964	0.255	0.020	0.024	54.1
0.283	0.075	0.008	0.012	51.2
0.000	0.000	0.00	0.00	

[2] The data tabulated was rounded to three decimals. The actual data was recorded to five decimals through a computer operated calibration system.

Appendix C

Flow Measurement Interpretation (Curve Fitting)

In order to use the calibration result to interpret flow rate measured in terms of voltage, curve fitting to the calibration result was necessary. The best fit equations for the two flow transducers were obtained respectively using natural logarithms and 6th order polynomials, which minimized the deviation (or fitting error) from the actual flow rate in terms of average and maximum fitting errors. The fitting did not include the two smallest calibration points (refer to Table B.1 in Appendix B) as they were not reliable. The curve fitting equations are:

For flow meter 8710157:

$$y = -0.0002x^6 - 0.0006x^5 + 0.0005x^4 + 0.0085x^3 - 0.0073x^2 + 0.5063x + 2.4521 \quad \text{(C-1)}$$

For flow meter 8710158:

$$y = -0.0004x^6 - 0.001x^5 + 0.0033x^4 + 0.0112x^3 - 0.0175x^2 + 0.4997x + 2.4730 \quad \text{(C-2)}$$

where x=ln(v), y=ln(q). v is the flow transducer outputs in volt, and q is flow rate in l/min.

The fitting results are summarized in Table C.1 (for the flow meter 8710157) and Table C.2 (for the flow meter 8710158) against actual measurement. When the output voltage was less than 0.01 voltage, the interpreted flow rate was forced to be zero.

Table C.1 Curve Fitting for the flow meter 8710157

Volts.(Record) 8710157	Q (l/min) (Record)	Q (GPM) (Record)	Predicted Q (l/min)	Abs. Deviation (l/min)
10.19	38.256	10.106	37.985	0.2707
10.00	37.884	10.008	37.652	0.2326
9.13	36.178	9.557	36.049	0.1290
8.12	33.982	8.977	34.025	0.0432
7.07	31.644	8.359	31.721	0.0770
6.20	29.613	7.823	29.643	0.0301
5.72	28.403	7.503	28.425	0.0223
5.33	27.393	7.237	27.395	0.0019
4.94	26.307	6.950	26.326	0.0192
4.27	24.464	6.463	24.389	0.0750
3.75	22.835	6.032	22.788	0.0474
3.25	21.229	5.608	21.151	0.0782
2.77	19.542	5.162	19.473	0.0695
2.15	17.173	4.537	17.101	0.0712
1.82	15.690	4.145	15.713	0.0229
1.52	14.334	3.787	14.346	0.0114
1.19	12.609	3.331	12.680	0.0703
1.02	11.697	3.090	11.730	0.0328
0.86	10.766	2.844	10.757	0.0090
0.69	9.590	2.533	9.610	0.0201
0.56	8.575	2.265	8.624	0.0487
0.38	7.035	1.859	7.018	0.0175
0.27	5.858	1.548	5.815	0.0440
0.17	4.541	1.200	4.455	0.0862
0.134	3.872	1.023	3.860	0.0121

(To be continued)

Table C.1 Curve Fitting for the flow meter 8710157 (continued)

Volts.(Record) 8710157	Q (l/min) (Record)	Q (GPM) (Record)	Predicted Q (l/min)	Abs. Deviation (l/min)
0.116	3.472	0.917	3.530	0.0582
0.036	1.615	0.427	1.508	0.1072
0.020	0.964	0.255	0.819	0.1441
0.008	0.283	0.075	0.164	0.1193
0.000	0.000	0.000	0.0	0.0
			Ave.:	0.067971
			Max.:	0.27072

Table C.2 Curve fitting for the flow meter 8710158

Volts. (Record) 8710158	Q (l/min) (Record)	Q (GPM) (Record)	Predicted Q (l/min)	Abs. Deviation (l/min)
10.19	38.256	10.106	38.269	0.0128
10.00	37.884	10.008	37.939	0.0545
9.10	36.178	9.557	36.282	0.1042
8.07	33.982	8.977	34.192	0.2107
7.00	31.644	8.359	31.796	0.1519
6.14	29.613	7.823	29.696	0.0826
5.66	28.403	7.503	28.453	0.0497
5.27	27.393	7.237	27.403	0.0102
4.89	26.307	6.950	26.345	0.0385
4.22	24.464	6.463	24.387	0.0770
3.71	22.835	6.032	22.808	0.0267
3.21	21.229	5.608	21.172	0.0571
2.72	19.542	5.162	19.465	0.0771
2.11	17.173	4.537	17.146	0.0269

(To be continued)

Table C.2 Curve fitting for the flow meter 8710158 (Continued)

Volts. (Record) 8710158	Q (l/min) (Record)	Q (GPM) (Record)	Predicted Q (l/min)	Abs. Deviation (l/min)
1.77	15.690	4.145	15.721	0.0302
1.47	14.334	3.787	14.348	0.0139
1.14	12.609	3.331	12.657	0.0476
0.98	11.697	3.090	11.739	0.0419
0.82	10.766	2.844	10.730	0.0357
0.66	9.590	2.533	9.599	0.0090
0.54	8.575	2.265	8.640	0.0647
0.37	7.035	1.859	7.041	0.0051
0.26	5.858	1.548	5.776	0.0829
0.17	4.541	1.200	4.517	0.0237
0.131	3.872	1.023	3.872	0.0001
0.114	3.472	0.917	3.562	0.0907
0.038	1.615	0.427	1.686	0.0705
0.024	0.964	0.255	1.071	0.1072
0.012	0.283	0.075	0.337	0.0537
0.00	0.000		0	0.0000
			Ave :	0.0571
			Max.:	0.2107