Identification and Control of Nonlinear Systems using Multiple Models

BY

LAI CHOW YIN

A THESIS SUBMITTED
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
NUS GRADUATE SCHOOL FOR INTEGRATIVE
SCIENCES AND ENGINEERING
NATIONAL UNIVERSITY OF SINGAPORE
2011

Abstract

Most of the systems in our real life are inherently nonlinear. One simple way to control a nonlinear plant over a large operating region is by utilizing the "divide and conquer" strategy. A few operating points which cover the whole range of system's operation are chosen, and a linear approximation is obtained at each of these operating points. The designer then designs one local controller for each local model, and activates one of these local controllers when the process is operating in the neighborhood of the corresponding linearization point. This is the basic idea behind the gain scheduling approach, supervisory control and multiple model control, which have found popularity in the industry as well as in flight control.

The aim of our work is to design multiple model controllers for nonlinear systems for tracking purposes. To this aim, the nonlinear system is approximated as piecewise affine autoregressive system with exogenous inputs (PWARX). Firstly, we propose a general framework for the identification of discrete-time time-varying system, where both offline and online identification algorithms for linear as well as nonlinear systems can be derived. Built upon this work, we further propose a simple and efficient algorithm which can automatically provide accurate PWARX models of nonlinear systems based on measured input-output data. The proposed algorithm is shown to be robust against noise as well as uncertainties in the model order. Next, we move on to design the local controllers based on the obtained PWARX model, which are then patched together through switching to become a global controller for the nonlinear system. We provide a few solutions to deal with a causality issue whereby the determination of the active subsystem and the computation of control signal affect each other at the same time. The designed controllers show good performance both in simulation as well as in experimental studies. One issue related to the PWARX model identification is the number of subsystems to be used. We show that if the original piecewise affine system consists of N state space subsystems, then we will need more than N input-output subsystems to fully describe the system's behavior. We show via simulation studies that having the correct number of the input-output subsystems is crucial to obtain a good idenfication and control of piecewise affine system.

Acknowledgments

I would like to express my deepest appreciation to Prof. Tong Heng Lee and Assoc. Prof. Cheng Xiang for their inspiration, excellent guidance, support and encouragement. Their erudite knowledge and their deepest insights on the fields of control have made this research work a rewarding experience. I owe an immense debt of gratitude to them for having given me the curiosity about the learning and research in the domain of control. Also, their rigorous scientific approach and endless enthusiasm have influenced me greatly. Without their kindest help, this thesis and many others would have been impossible.

Thanks also go to NUS Graduate School for Integrative Sciences and Engineering in National University of Singapore, for the financial support during my pursuit of a PhD.

I would like to thank Assoc. Prof. Abdullah Al Mamun, Prof. Ben Mei Chen and Prof. Shuzhi Sam Ge at the National University of Singapore, Prof. Frank Lewis at the University of Texas at Arlington, Prof. Masayoshi Tomizuka and Dr. Kyoungchul Kong at the University of California at Berkeley, and Dr. Venkatakrishnan Ventakaraman at the Data Storage Institute of Singapore who provided me kind encouragement and constructive suggestions for my research. I am also grateful to all my friends in Control and Simulation Lab, National University of Singapore. Their kind assistance and friendship have made my life in Singapore easy and colorful.

Last but not least, I would thank my family members for their support, understanding, patience and love during past several years. This thesis, thereupon, is dedicated to them for their infinite stability margin.

Contents

List of Figures

List of Tables

Nomenclature

Symbol	Meaning or Operation
$\triangleq$	Definition
$\in$	In the set
$\cap$	Intersection of sets
$\cup$	Union of sets
$\bar{(\cdot)}$	Mean value
$\hat{(\cdot)}$	Estimated value
$(\cdot)^*$	Optimal value
$\lvert \cdot \rvert$	Cardinality of set

Symbol	Meaning or Operation
$e(t)$	Identification error
k	Iteration in the parameter identification algorithm
m	The m^{th} subsystem
N	Number of subsystems
$N(\bar{x}, \sigma_\epsilon^2)$	White noise with mean $\bar{x}$ and variance σ_ϵ^2
n	Order of regression vector
n_a	Order of y in the regression vector
n_b	Order of u in the regression vector
p	Order of state space model
$\mathbf{R}$	Set of real numbers
$r(t)$	reference signal
T	Number of data observations
t	Sampling times
$u(t)$	Control input
$x(t)$	Observation data
X	Regressor space
$\tilde{X}$	Modified regressor space
X_i	The i^{th} partition of the regressor space
$\tilde{X}_i$	The i^{th} partition of the modified regressor space
$y(t)$	Process output

Symbol	Meaning or Operation
δ	Local gradient within the multilayer perceptrons
Δ_θ	Quality measure for parameter identification
ϵ	Bound of identification error
$\eta(t)$	Measurement noise
θ	Parameter
λ	Weight on the penalty of control size
$\xi(t)$	State variables
$\sigma(t)$	Switching signal
σ_ϵ^2	Variance of identification error
σ_n^2	Variance of measurement noise
$\varphi(x(t))$	Function of x(t)
$\varphi(t)$	Affine function of x(t)

Symbol	Meaning or Operation
ARX	Autoregressive systems with exogenous inputs
BPWA	Piecewise affine basis function
DWO	Direct weight optimization
EM	Expectation maximization
HDC	Hybrid decoupling constraints
HMM	Hidden Markov model
LGM	Least geometrical mean of error squares
LHM	Least harmonic mean of error squares
MILP	Mixed-integer linear programming
MIQP	Mixed-integer quadratic programming
MLD	Mixed logical dynamical systems
MLP	Multilayer perceptron
mp	Multiparametric programming
MPC	Model predictive control
NN	Neural networks
PWA	Piecewise affine
PWARX	Piecewise affine ARX
RPM	Revolution per minute
SSR	Sum of squared residues
SVM	Support vector machine
VAF	Variance accounted for

Chapter 1

Introduction

1.1 Background and motivation

The objective of "controlling a system" is to influence its behaviour so as to achieve a desired goal. Questions of control have been of great interest since ancient times, as can be seen from the design of self-regulating systems such as water clocks in antiquity and aqueducts in early Rome. In our modern society, control theory and application are assuming even more importance. Control mechanisms are ubiquitous in many systems ranging from Watt's steam engine governor in 1769 which ushered in the Industrial Revolution in England, to the sophisticated unmanned aircraft in our own times.

The fundamental concept of control is feedback. The three key elements of the feedback concept are measurement, comparison and adjustment. Firstly, the quantity of interest is measured using sensors. The measured value is then compared to the desired value, and the difference between these two is calculated. Finally, the process is adjusted to reduce or eliminate the error. The accustomed sequence of clause and effect in the above process is converted into a closed loop of interdependent events. This closed circle of information transmission, referred to as feedback, underlies the entire technology of automatic control based on self-regulation.

During the period 1932-1960 numerous methods were developed to control simple systems in an efficient manner. In particular, using both frequency domain as well as time domain methods based on pole-zero configurations of the relevant transfer functions, various design methods were developed for the control of linear systems described by difference or differential equations. These methods have been used extensively in the industry to design controllers for innumerable systems and have been found to be extremely robust and hence, very reliable. As a matter of fact, many control systems have already become an indispensable part in our daily life. For instance, the temperature regulation device of air conditioning system and the cruise control system of the automobile are both good examples of classical feedback controllers.

Advances in technology invariably call for faster and more accurate controllers. Although the afore-mentioned control methods have been rather successful so far, they rely on one key assumption that the system is linear, or at least sufficiently linear within a small range of operation. Real-world systems, on the other hand, are inherently nonlinear. In today's industry, systems and equipments are forced to work over a wider range of operation, resulting in the loss of validity of the linearity assumption. For example, chemical plants would use the same equipments to manufacture various products, each of which needs a different temperature or pressure as the optimal operating point. As such, a linear controller is likely to perform poorly, since the nonlinearity cannot be properly accounted for.

One possible solution to this problem is to design a nonlinear controller which can handle the system's nonlinearity and hence, works well in all operating regions. However, it is also well-known that this is a daunting task, since there is still no general methodology of nonlinear control design which can tackle all kinds of nonlinear systems.

Depending on the class and structure of the system, a particular technique (for e.g. feedback linearization, backstepping design etc.) could be more suitable than the others. Thus, the control engineer needs to have not only a deep understanding of the system itself, but also a collection of various alternatives of controller design methods. Another difficulty for the nonlinear controller design is that an accurate nonlinear model of the process, which is used to facilitate the controller design and the analysis of the closed loop system, is needed. To obtain accurate physical models of nonlinear systems can be very challenging, because the interplay among the mechanical, electrical, chemical, thermal or other properties of the system has to be properly understood.

1.2 Control of nonlinear systems using multiple models and piecewise affine models

Another simpler way to control a nonlinear system over a large operating region is by utilizing the "divide and conquer" strategy. A few operating points which cover the whole range of system's operation are chosen, and a linear approximation is obtained at each of these operating points. The designer then designs one local controller for each local model based on any of the well-known linear control design techniques, and activates one of these local controllers when the process is operating in the neighborhood of the corresponding linearization point. This is the basic idea behind the gain scheduling approach, supervisory control and multiple model control, which have found popularity in the industry as well as in flight control.

The scheme of this multiple model control approach is illustrated in Fig. 1.1, whereby the nonlinear process is drawn using dashed curve and the linear approximations over the whole nonlinearity range are sketched using solid lines. Note that we do not define

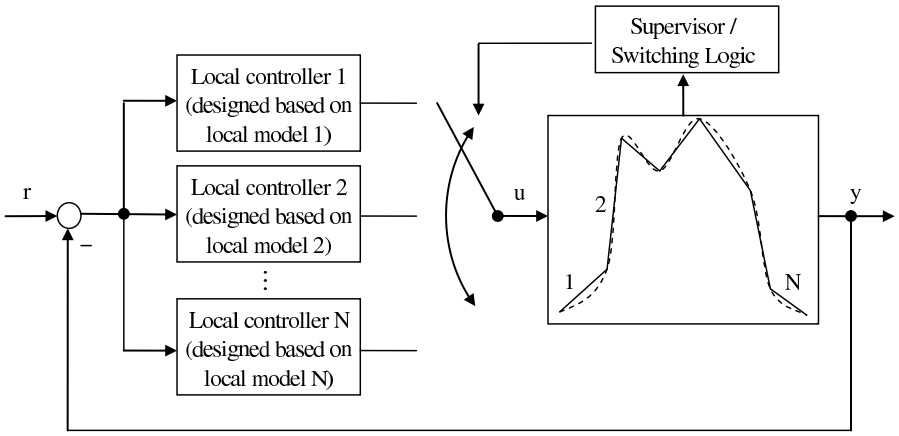

Figure 1.1: The multiple model control scheme

the physical meanings for the axes, i.e. the x-axis does not necessarily represent time. The dashed curve merely serves to illustrate a nonlinear system.

The use of multiple models is not new in control theory. In fact, the classical gain scheduling theory originates from the late 1950s in the field of aircraft control, in which the flight parameters are affected significantly by the altitude (dynamic pressure) and Mach number [37, 69]. Stein *et al.* [73] and Kallstrom *et al.* [31] used the gain scheduling approach for the control of F-8 aircraft and tankers, in 1977 and 1979 respectively. In recent years, Morse [47, 48] has been studying the use of multiple fixed models and optimization for robust set-point control. Narendra and Balakrishnan [54] proposed the idea of using multiple adaptive models and switching in order to improve the performance of an adaptive system while assuring stability. The works by Morse and by Narendra have gained a large following, and many theoretical as well as application papers have appeared since then.

The main advantage of the multiple model approach is that powerful linear design tools can be employed on difficult nonlinear problems. It is thus not surprising that this control scheme has been utilized to solve many real world control problems. Due to its

many advantages, we will follow similar approach in our work. However, contrary to most existing work whereby the controller switching is determined by comparing some cost functions of current and past errors, we propose to decide the active local controller directly using information regarding the location of the regression vector within the regressor space. This brings out the advantage of faster decision making. As for the terminology, a multiple-model system whose active linear/affine subsystem is determined by the location of the regression vector within the whole regressor space is called "piecewise affine system". Therefore, we can think of our approach as the "control of nonlinear systems using piecewise affine models", which is a special case of "control of nonlinear systems using multiple models".

1.3 Identification of nonlinear systems using piecewise affine autoregressive models with exogeneous inputs

As mentioned earlier, physical modeling of nonlinear systems is not always an easy task. Therefore, we are more in favor of using data-based modeling, i.e. to describe the system's characteristic through functions of measured input-output data. Many architectures for modeling nonlinear processes using measured data exist in the literature, for e.g. artificial neural networks, Hammerstein models, Wiener models, Volterra series and also piecewise affine ARX (AutoRegressive with eXogenous inputs) functions. Among these, the first four have the disadvantage that they are usually not affine in control, i.e. the control signal cannot be computed as a simple function of the reference and output directly by inverting the models.

Piecewise affine ARX models, on the other hand, have the distinct advantage that they are affine in control. As such, the control signal can be calculated easily by inverting

the model and then evaluate a function of the reference and the history of input-output signals. We shall therefore use piecewise affine ARX models to approximate the nonlinear functions. Another reason for using piecewise affine ARX models to model nonlinear functions is that this model structure automatically fits into the framework of multiple model control strategy, since each of the ARX subsystems of the piecewise affine ARX model represents one local model of the nonlinear system.

The literature on piecewise affine approximation of nonlinear systems appeared since the 1970s (see [72] and the references therein). It was mentioned that it is worthwhile to investigate piecewise linear(affine) models due to "simplicity of implementation, theoretical analysis and calculation". In recent years, there is a growing interest in the identification of piecewise affine systems. We will defer the description of various identification methods to the relevant chapter, along with our own proposed algorithm which shows many advantages over the other existing methods.

1.4 Objectives and Contributions

Our main objective is to control nonlinear systems for reference tracking. To this aim, we first identify the nonlinear system using piecewise affine ARX models, and then calculate the control signals based on the identified models. The contributions of our work are as follows.

1.4.1 Development of a new general framework for the identification of time-varying systems

Our first contribution, detailed in Chapter 2, is the development of a new general framework for the identification of discrete-time time-varying systems. Firstly, to make the problem tractable, we assume that the time-variation can be approximated by a

piecewise-constant function assuming a finite number N of unknown values. Thus, the change in the system parameters is equivalent to switching from one subsystem to another, and the system can be modelled as a switching system

$$y(t+1) = \begin{cases} \varphi^T(x(t))\theta_1^* & \text{if subsystem } 1 \\ \vdots & \vdots \\ \varphi^T(x(t))\theta_m^* & \text{if subsystem } m \in (1,\ldots,N) \\ \vdots & \vdots \\ \varphi^T(x(t))\theta_N^* & \text{if subsystem } N \end{cases} \tag{1.1}$$

if the underlying subsystems are linear or linear in the parameters, and

$$y(t+1) = \begin{cases} f_1(x(t),\theta_1^*) & \text{if subsystem } 1 \\ \vdots & \vdots \\ f_m(x(t),\theta_m^*) & \text{if subsystem } m \in (1,\ldots,N) \\ \vdots & \vdots \\ f_N(x(t),\theta_N^*) & \text{if subsystem } N \end{cases} \tag{1.2}$$

if the underlying subsystems are nonlinear.

Our proposed algorithm is able to provide the parameters of the linear (or linear in the parameters) subsystems θ_m^*, as well as the nonlinear functions $f_m(x(t),\theta_m^*)$, without needing to know how the system switches. Furthermore, the algorithms can be easily extended to cater for both offline and online identification of the parameters. The applicability of our method on all these possibilities, i.e. identification of linear and nonlinear subsystems as well as derivation of offline and online methods, motivated us to name it "the general framework for the identification of time-varying systems".

1.4.2 Identification of nonlinear systems using piecewise affine autoregressive models with exogeneous inputs

In Chapter 3, we proceed to identify a piecwise affine ARX (PWARX) system with N partitions

$$y(t+1) = \begin{cases} \varphi^T(t)\theta_1^* & \text{if } x(t) \in X_1 \\ \vdots & \vdots \\ \varphi^T(t)\theta_m^* & \text{if } x(t) \in X_{m\in(1,\ldots,N)} \\ \vdots & \vdots \\ \varphi^T(t)\theta_N^* & \text{if } x(t) \in X_N \end{cases} \tag{1.3}$$

where

$$x(t) = [y(t), \ldots, y(t-n_a), u(t), \ldots, u(t-n_b)]^T \tag{1.4}$$

$$\varphi(t) = \left[x^T(t), 1\right]^T \tag{1.5}$$

and

$$\cup_{i=1}^{N} X_i = X \text{ and } X_i \cap X_j = \emptyset, \forall i \neq j \tag{1.6}$$

where X denotes the whole regressor space. For this purpose, both the parameters of the affine subsystems θ_m^* as well as the partition of the regressor space X_m have to be estimated. The first task, i.e. identification of the subsystems' parameters, follows exactly the same procedure which is detailed in the general framework for the identification of time-varying system (Chapter 2). The procedure to estimate the partition of the regressor space is provided in Chapter 3.

We then propose to fit the input-output data of nonlinear systems using the PWARX model. Both simulation and experimental studies validated the efficacy of our algorithms in identifying nonlinear systems using piecewise affine approximations.

One novelty in this part of the work is the introduction of the "modified regressor space $\tilde{\mathrm{X}}_{\mathrm{m}}$, where the control signal at the most current step, $u(t)$, is omitted from the

original regressor space $X(t)$. This modification to the regression vector will simplify the controller design at the later stage, and provide significant improvement to the control performance.

1.4.3 Control of nonlinear systems using piecewise affine autoregressive models with exogeneous inputs

Having obtained the PWARX model of the nonlinear system, we then move on to design controllers for tracking purposes. For each of the local models, we design a one-step-ahead controller which is essentially a model-inversion controller, and switch among the controllers for different operating regions. The details of this part of the works are provided in chapter 4.

If we follow the conventional definition of the regressor space of a piecewise affine system, there will exist one akward chicken-and-egg situation whereby the determination of the active subsystem as well as the computation of the control signal are interlaced, i.e. both need the information from each other at the same time. Our contribution lies in providing a few different solutions to this problem, and showing through simulation as well as experimental studies that the control performance is truly satisfactory.

1.4.4 Input-output models of switching state space systems

In the earlier part of the work, the number N of subsystems of the PWARX model is assumed known, or was determined via a trial-and-error process by observing the fitting result for training data and test data. For the first case, there is actually a difference between the number N whether the subsystems are described using input-output equations, or if they are described using state space models. We show in this work that if there are N subsystems, each of which is a state space model of order p, then

there exist a total of N^p input-output subsystems. Therefore, defining N^p input-output subsystems is crucial for accurate identification and control of a switching piecewise affine system with N state space models.

Our contribution is that we rigorously develop a rather simple, yet effective, procedure to derive the input-output models from switching (piecewise affine) state space models. We call the additional input-output models during switching as the "transition models". We further prove that the models are invariant to coordinate transformations of the states. The advantage of our approach is its relative simplicity (and thus an easily adoptable methodology), and its ready applicability for the typically more difficult classes of switching nonlinear systems and MIMO systems. The proposed approach is detailed in chapter 5.

Finally, we provide a list of possible future research directions in this field. These are given in chapter 6.

Chapter 2

A General Framework for Least-Squares Based Identification of Time-Varying Systems using Multiple Models

2.1 Introduction

The multiple model adaptive control [11, 15, 16, 21, 47, 48, 54, 56, 57, 84] is an effective strategy in improving the performance of an adaptive system while assuring stability. One important element of the multiple model adaptive control strategy is the set of fixed models, which should be built using some *a priori* knowledge of the plant. If one of the fixed models is close to the true plant in some sense, then the transient performance of the adaptive system will be improved using the multiple model approach. The question we would like to address in this chapter is how to locate these models in a rapidly time-varying system.

To make the problem tractable, the time-variation is approximated by a piecewise-constant function assuming a finite number N of unknown values [19]. Thus, the change

in the system parameters is equivalent to switching from one subsystem to another. If the switching signal is available to the observer, then the observation data can be segmented into different subsets, and the standard least squares can be applied to identify the subsystems using the subsets. However, if the data are mixed and the underlying switching signal is not available, it is a very challenging problem to identify the parameters of the subsystems.

To successfully identify the parameters of the underlying subsystems, an algorithm should be able to perform clustering (segmentation), regression and optimization. In [77], Vidal *et al* proposed an algebraic geometric approach to identify a class of linear hybrid systems, which is then extended for recursive identification of switched ARX systems in [78]. The problem with this algorithm is that it is quite vulnerable to noise, and that it is not easily extended for identification of nonlinear systems. The Expectation Maximization (EM) [18] based algorithm [24] and the identification algorithm proposed in [19] can also be used for the identification of switched linear system. Although simulation results show that the parameter estimates can converge, the mathematical convergence analysis is not readily established. Also, only linear subsystems were considered in [19] and [24]. In [61], the EM based algorithm is used to train the Radial-Basis-Function networks to identify switched system with nonlinear subsystems. However, a low switching rate assumption was imposed in [61].

In this chapter, we would like to propose a general framework for the identification of discrete-time time-varying system, which would be applicable for switched system with linear and nonlinear subsystems. The general framework also facilitates the derivation of both offline and online parameter identification algorithms. Extensive simulation studies show that our algorithm can indeed provide accurate estimates of the plant parameters

even in noisy cases. Also, a preliminary convergence analysis is available.

This chapter is organized as follows. In section 2.2, the weighted least squares are reviewed. Section 2.3 introduces our proposed general framework for the identification of time-varying system. In section 2.4, a multiple model based weighted least squares algorithm is derived for offline identification of switched system with linear subsystems. The online identification counterpart, which is based on gradient descent algorithm, is derived in section 2.5. Section 2.6 deals with the training of multiple multi-layer-perceptrons for identification of switched system with nonlinear subsystems. Finally, in section 2.7, conclusions will be drawn.

2.2 Mathematical Preliminaries

Since the proposed procedure hinges on transforming certain cost functions into weighted least squares, a brief summary of the weighted least squares algorithm is given in this section.

Pairs of observations $\{(x(i), y(i+1)), i = 1, 2, \ldots, T\}$ are obtained from an experiment. Assume that the data is fitted by a model in the form of

$$\hat{y}(x, \alpha) = \varphi_1(x)\alpha_1 + \varphi_2(x)\alpha_2 + \cdots + \varphi_n(x)\alpha_n = \varphi^T(x)\alpha \tag{2.1}$$

where $\hat{y}$ denotes the estimated value for the true y, $\varphi(x) = (\varphi_1(x), \varphi_2(x), \ldots,$ $\varphi_n(x))^T$ is the regression vector, and $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n)^T$ is the unknown parameter vector to be estimated. Then, the weighted least squares is to minimize the following cost function

$$\begin{aligned} J_w(\alpha) &= \sum_{i=1}^{T} (w(i)(y(i+1) - \hat{y}(i+1)))^2 \\ &= \sum_{i=1}^{T} (w(i)y(i+1) - w(i)\varphi^T(x(i))\alpha)^2 \end{aligned} \tag{2.2}$$

where $w(i)$ weighs the influence of the data point $(x(i), y(i+1))$ over the parameter estimates.

If the data is sufficiently rich, the unique solution that minimizes the cost function in (2.2) is

$$\alpha^* = (\Phi_w^T \Phi_w)^{-1} \Phi_w^T Y_w \tag{2.3}$$

where the weighted output vector is

$$Y_w = (w(1)y(2), w(2)y(3), \ldots, w(T)y(T+1))^T \tag{2.4}$$

and the weighted regression matrix is

$$\Phi_w = \begin{pmatrix} w(1)\varphi^T(x(1)) \\ \vdots \\ w(T)\varphi^T(x(T)) \end{pmatrix} \tag{2.5}$$

2.3 General Framework: Multiple Model Based Least Squares

As mentioned in the introduction, the time-varying discrete-time process is approximated by a switching system where each subsystem describes the dynamics of the system under certain environment. Such a general switching system can be described through the following equation

$$y(t+1) = \begin{cases} f_1(x(t), \theta_1^*) & \text{if subsystem 1} \\ \vdots & \vdots \\ f_m(x(t), \theta_m^*) & \text{if subsystem } m \in (1, \ldots, N) \\ \vdots & \vdots \\ f_N(x(t), \theta_N^*) & \text{if subsystem } N \end{cases} \tag{2.6}$$

where N denotes the number of subsystems, and $f_m(x(t), \theta_m^*)$ represents the possibly nonlinear function of the regressor $x(t)$ and parameters θ_m^*. If the function $f_m(x(t), \theta_m^*)$

is linear or linear in the parameters, then the switched system can be written as

$$y(t+1) = \begin{cases} \varphi^T(x(t))\theta_1^* & \text{if subsystem } 1 \\ \vdots & \vdots \\ \varphi^T(x(t))\theta_m^* & \text{if subsystem } m \in (1,\ldots,N) \\ \vdots & \vdots \\ \varphi^T(x(t))\theta_N^* & \text{if subsystem } N \end{cases} \tag{2.7}$$

If the number of the subsystems N is finite and if the switching signal is known, then the parameter identification problem can be easily solved, as the data can be easily separated into different groups (subsystems), and the parameters of each individual subsystem can be identified in a straightforward manner using standard identification method. However, if the switching signal is unknown, the parameter identification becomes a formidable task.

In this chapter, we will present a general framework for the identification of discrete-time time-varying system, which can be applied on switched system with linear or nonlinear subsystems, without needing to know the switching signal. The general framework also enables the derivation of both offline and online parameter identification algorithms. The method is as follows.

2.3.1 The Cost Functions

The first question is how to construct a cost function such that the model parameter vectors $\theta_1^*, \ldots, \theta_N^*$ would be the solutions for the global minimum (zero for noiseless data). Among many possible candidates for the cost function to satisfy this requirement, two effective cost functions are introduced in our work: 1) the geometrical mean of the squares of errors

$$J_g(\theta_1,\ldots,\theta_m,\ldots,\theta_N) = \frac{1}{2}\sum_{i=1}^{T} e_1^2(i)\ldots e_m^2(i)\ldots e_N^2(i) \tag{2.8}$$

and 2) the harmonic mean of the squares of errors

$$J_h(\theta_1, \ldots, \theta_m, \ldots, \theta_N) = \frac{1}{2} \sum_{i=1}^{T} \frac{1}{\frac{1}{e_1^2(i)} + \cdots + \frac{1}{e_m^2(i)} + \cdots + \frac{1}{e_N^2(i)}} \tag{2.9}$$

where $e_m(i)$ is the identification error of the m^{th} model

$$e_m(i) = y(i+1) - f_m(x(i), \theta_m) \tag{2.10}$$

If any of the parameter estimates $\theta_1, \ldots, \theta_N$ equals the true value θ_m^* at every time instant, where m can be any number in the set $\{1, \ldots, N\}$ due to switching, then the corresponding identification error $e_m(i)$ will be zero. This will render the cost functions (2.8) and (2.9) zero, i.e. the proposed cost functions share the same optimal solution. However, they have different properties (e.g. sensitivity to noise, computational load and convergence property) and thus should be selected according to the characteristics of applications. In the following section, the optimization processes and analyses on both cost functions (2.8) and (2.9) are introduced.

Remark 2.1 *Although the exact definition of geometrical mean involves taking the roots of the product of all the arguments, we still term the cost function (2.8) as the geometrical mean squares.*

2.3.2 A New Perspective on the Cost Functions

Now the next question is how to minimize the cost function. Since the cost functions are no longer quadratic functions of the parameters, there is no direct way to find out the global minimum solution using the conventional optimization algorithms such as gradient descent and conjugate gradient methods.

In the following, we would like to propose a new perspective on the cost functions, which would enable us to use well-known algorithms with only slight modification to

identify the parameters of the subsystems. This new perspective is inspired by the well-known EM Algorithm [18], which deals with the incomplete-data problem by solving the optimization problem recursively as if the unknown parameters were the true values at each iteration.

It is observed that if we fix all of the parameters in (2.8) except the one we would like to estimate at a particular moment, then the cost function becomes quadratic with respect to the parameter

$$\begin{aligned} J_g(\theta_1, \ldots, \theta_m, \ldots, \theta_N) &= \frac{1}{2} \sum_{i=1}^{T} \underbrace{e_1^2(i) \ldots}_{\text{fixed}} e_m^2(i) \underbrace{\ldots e_N^2(i)}_{\text{fixed}} \\ &= \frac{1}{2} \sum_{i=1}^{T} w_m^2(i) e_m^2(i) \end{aligned} \tag{2.11}$$

where

$$w_m(i) = \prod_{\substack{j=1 \\ j \neq m}}^{N} e_j(i) \tag{2.12}$$

Similarly, the cost function (2.9) can be transformed into

$$J_h(\theta_1, \ldots, \theta_m, \ldots, \theta_N) = \frac{1}{2} \sum_{i=1}^{T} w_m^2(i) e_m^2(i) \tag{2.13}$$

by fixing

$$w_m^2(i) = \frac{\dfrac{1}{e_m^2(i)}}{\dfrac{1}{e_1^2(i)} + \cdots + \dfrac{1}{e_m^2(i)} + \cdots + \dfrac{1}{e_N^2(i)}} \tag{2.14}$$

From this perspective, the geometric mean squares (2.8) and the harmonic mean squares (2.9) become the usual weighted least squares (2.11) and (2.13) respectively. It is this change of perspective which enables us to derive various algorithms for offline and online identification of switched systems with linear or nonlinear subsystems. We will defer the details of the learning algorithms to the subsequent sections.

2.4 Offline Identification of Linear Time-Varying Systems

First of all, we will consider the offline identification of linear time-varying systems, which are assumed to be describable by equation (2.7). For the convenience of the reader, the system equation is repeated here:

$$y(t+1) = \begin{cases} \varphi^T(x(t))\theta_1^* & \text{if subsystem } 1 \\ \vdots & \vdots \\ \varphi^T(x(t))\theta_m^* & \text{if subsystem } m \in (1,\ldots,N) \\ \vdots & \vdots \\ \varphi^T(x(t))\theta_N^* & \text{if subsystem } N \end{cases} \tag{2.15}$$

2.4.1 The Least Geometric Mean Squares

To identify the parameters of the N subsystems in an offline manner, we propose to minimize the cost function (2.8) iteratively. Assume the parameter vectors at the k^{th} iteration are $\theta_1(k),\ldots,\theta_m(k),\ldots,\theta_N(k)$, then the cost function is

$$\begin{aligned} J_g(k) &= J_g(\theta_1(k),\ldots,\theta_m(k),\ldots,\theta_N(k)) \\ &= \frac{1}{2}\sum_{i=1}^{T} e_{1,k}^2(i)\ldots e_{m,k}^2(i)\ldots e_{N,k}^2(i) \end{aligned} \tag{2.16}$$

where $e_{m,k}(i)$ is the identification error of the m^{th} models at the k^{th} step

$$e_{m,k}(t) = y(t+1) - \varphi^T(x(t))\theta_m(k) \tag{2.17}$$

Following the discussion in section 2.3.2, we can view (2.16) as a weighted least squares cost function. We are going to propose one simple algorithm such that the value of the cost function is guaranteed to decrease at each iteration. The N parameter sets can be estimated separately in N steps in each iteration as follows.

Firstly, the parameter $\theta_1(k+1)$ is computed by minimizing (2.16) assuming the parameters of all the other models, $\theta_2(k),\ldots,\theta_N(k)$, are fixed. Then the cost function

can be rewritten as

$$\begin{aligned} J_g(\theta_1) &= \frac{1}{2}\sum_{t=1}^{T}(y(t+1)-\varphi^T(x(t))\theta_1)^2 e_{2,k}^2(t)\dots e_{N,k}^2(t) \\ &= \frac{1}{2}\sum_{t=1}^{T}(y(t+1)-\varphi^T(x(t))\theta_1)^2 w_{1,k}^2(t) \\ &= \frac{1}{2}\sum_{t=1}^{T}(w_{1,k}(t)y(t+1)-w_{1,k}(t)\varphi^T(x(t))\theta_1)^2 \end{aligned} \tag{2.18}$$

where

$$w_{1,k}(t) = \prod_{j=2}^{N} e_{j,k}(t) \tag{2.19}$$

Following (2.3) for the weighted least squares, and by assuming that the data are sufficiently rich, the unique optimal solution can be obtained as

$$\theta_1^* = (\Phi_{1,k}^T\Phi_{1,k})^{-1}\Phi_{1,k}^T Y_{1,k} \tag{2.20}$$

where

$$Y_{1,k} = (w_{1,k}(1)y(2), w_{1,k}(2)y(3), \dots, w_{1,k}(T)y(T+1))^T \tag{2.21}$$

$$\Phi_{1,k} = \begin{pmatrix} w_{1,k}(1)\varphi^T(x(1)) \\ \vdots \\ w_{1,k}(T)\varphi^T(x(T)) \end{pmatrix} \tag{2.22}$$

Thus let

$$\theta_1(k+1) = (\Phi_{1,k}^T\Phi_{1,k})^{-1}\Phi_{1,k}^T Y_{1,k} \tag{2.23}$$

The second step is to update the parameters of the second model assuming the parameters of all the other models are fixed. Note: it is better to use the latest update of $\theta_1(k+1)$ rather than the previous one $\theta_1(k)$. For this purpose, we need to re-calculate the identification error of the first model as

$$e_{1,k+1}(t) = y(t+1) - \varphi^T(x(t))\theta_1(k+1) \tag{2.24}$$

Then the corresponding cost function is

$$\begin{aligned} J_g(\theta_2) &= \frac{1}{2}\sum_{t=1}^{T}(y(t+1)-\varphi^T(x(t))\theta_2)^2 e_{1,k+1}^2(t)e_{3,k}^2(t)\dots e_{N,k}^2(t) \\ &= \frac{1}{2}\sum_{t=1}^{T}(y(t+1)-\varphi^T(x(t))\theta_2)^2 w_{2,k}^2(t) \\ &= \frac{1}{2}\sum_{t=1}^{T}(w_{2,k}(t)y(t+1)-w_{2,k}(t)\varphi^T(x(t))\theta_2)^2 \end{aligned} \tag{2.25}$$

where

$$w_{2,k} = e_{1,k+1}(t)e_{3,k}(t)\dots e_{N,k}(t) \tag{2.26}$$

The optimal solution to this problem is then

$$\theta_2^* = (\Phi_{2,k}^T\Phi_{2,k})^{-1}\Phi_{2,k}^T Y_{2,k} \tag{2.27}$$

where

$$Y_{2,k} = (w_{2,k}(1)y(2), w_{2,k}(2)y(3), \dots, w_{2,k}(T)y(T+1))^T \tag{2.28}$$

$$\Phi_{2,k} = \begin{pmatrix} w_{2,k}(1)\varphi^T(x(1)) \\ \vdots \\ w_{2,k}(T)\varphi^T(x(T)) \end{pmatrix} \tag{2.29}$$

Thus let

$$\theta_2(k+1) = (\Phi_{2,k}^T\Phi_{2,k})^{-1}\Phi_{2,k}^T Y_{2,k} \tag{2.30}$$

Similarly, the m^{th} step is to update the parameter of the m^{th} model assuming the parameters of all the other models are fixed. Recall that it is better to use the latest update of $\theta_1(k+1)\dots\theta_{m-1}(k+1)$ rather than the previous one:

$$\theta_m(k+1) = (\Phi_{m,k}^T\Phi_{m,k})^{-1}\Phi_{m,k}^T Y_{m,k} \tag{2.31}$$

where

$$Y_{m,k} = (w_{m,k}(1)y(2), w_{m,k}(2)y(3), \dots, w_{m,k}(T)y(T+1))^T \tag{2.32}$$

$$\Phi_{m,k} = \begin{pmatrix} w_{m,k}(1)\varphi^T(x(1)) \\ \vdots \\ w_{m,k}(T)\varphi^T(x(T)) \end{pmatrix} \tag{2.33}$$

$$w_{m,k}(t) = \left(\prod_{j=1}^{m-1} e_{j,k+1}(t)\right)\left(\prod_{j=m+1}^{N} e_{j,k}(t)\right) \tag{2.34}$$

Since at each step, the resulting parameter corresponds to the unique global minimum point, we have

$$J_g(\theta_1(k+1), \theta_2(k), \ldots, \theta_N(k)) \leq J_g(\theta_1(k), \theta_2(k), \ldots, \theta_N(k))$$

$$J_g(\theta_1(k+1), \theta_2(k+1), \ldots, \theta_N(k)) \leq J_g(\theta_1(k+1), \theta_2(k), \ldots, \theta_N(k))$$

$$\vdots$$

$$\begin{aligned} &J_g(\theta_1(k+1), \theta_2(k+1), \ldots, \theta_m(k+1), \ldots \theta_N(k)) \\ &\quad \leq J_g(\theta_1(k+1), \theta_2(k+1), \ldots, \theta_m(k), \ldots \theta_N(k)) \end{aligned} \tag{2.35}$$

$$\vdots$$

Therefore, the cost function at each iteration is guaranteed to decrease monotonically, such that

$$J_g(\theta_1(k+1), \theta_2(k+1), \ldots, \theta_N(k+1)) \leq J_g(\theta_1(k), \theta_2(k), \ldots, \theta_N(k)) \tag{2.36}$$

Thus, the cost function is guaranteed to converge. But does it converge to zero, the global minimum? While extensive simulation studies indicate that it indeed converges to zero when the data is rich, the mathematical proof is lacking at the moment. Furthermore, the meaning of richness of data is still vague at the moment. For successful identification using multiple models, it does not suffice for the input signal to be persistently exciting. Intuitively, each subsystem must also be visited frequent enough. Formalizing this intuition mathematically will be part of the future work.

2.4.2 The Least Harmonic Mean Squares

An algorithm to minimize the harmonic mean squares (2.9) can be derived similarly. Assume that the parameter vectors at the k^{th} iteration are $\theta_1(k), \ldots, \theta_m(k), \ldots, \theta_N(k)$, then the cost function is

$$\begin{aligned} J_h(k) &= J_h(\theta_1(k), \ldots, \theta_m(k), \ldots, \theta_N(k)) \\ &= \frac{1}{2}\sum_{t=1}^{T} \frac{1}{\frac{1}{e_{1,k}^2(t)} + \cdots + \frac{1}{e_{m,k}^2(t)} + \cdots + \frac{1}{e_{N,k}^2(t)}} \end{aligned} \tag{2.37}$$

where

$$e_{m,k}(t) = y(t+1) - \varphi^T(t)\theta_m(k) \tag{2.38}$$

Firstly, the parameter $\theta_1(k+1)$ is calculated by minimizing:

$$\begin{aligned} J_h(\theta_1) &= \frac{1}{2}\sum_{t=1}^{T} \frac{1}{\frac{1}{e_{1,k}^2(t)} + \cdots + \frac{1}{e_{N,k}^2(t)}} \\ &= \frac{1}{2}\sum_{t=1}^{T} \underbrace{\frac{\frac{1}{e_{1,k}^2(t)}}{\frac{1}{e_{1,k}^2(t)} + \cdots + \frac{1}{e_{N,k}^2(t)}}}_{w_{1,k}^2(t)} e_{1,k}^2(t) \\ &= \frac{1}{2}\sum_{t=1}^{T} w_{1,k}^2(t)(y(t+1) - \varphi^T(t)\theta_1)^2 \\ &= \frac{1}{2}\sum_{t=1}^{T} (w_{1,k}(t)y(t+1) - w_{1,k}(t)\varphi^T(t)\theta_1)^2 \end{aligned} \tag{2.39}$$

By assuming that the data are sufficiently rich, the unique optimal solution can be obtained as

$$\theta_1^* = (\Phi_{1,k}^T \Phi_{1,k})^{-1} \Phi_{1,k}^T Y_{1,k} \tag{2.40}$$

where

$$Y_{1,k} = (w_{1,k}(1)y(2), w_{1,k}(2)y(3), \ldots, w_{1,k}(T)y(T+1))^T \tag{2.41}$$

$$\Phi_{1,k} = \begin{pmatrix} w_{1,k}(1)\varphi^T(1) \\ \vdots \\ w_{1,k}(T)\varphi^T(T) \end{pmatrix} \tag{2.42}$$

Thus let

$$\theta_1(k+1) = (\Phi_{1,k}^T \Phi_{1,k})^{-1} \Phi_{1,k}^T Y_{1,k} \tag{2.43}$$

Notice that in (2.39), the weight $w_{1,k}$ actually contains the term $\theta_1(k)$. Nevertheless, it is left untouched within the weight when $\theta_1(k+1)$ is updated according to (2.43). This makes the convergence proof of the algorithm harder to establish, compared to the least geometrical mean squares counterpart.

Next, the parameter $\theta_2(k+1)$ is calculated by minimizing:

$$\begin{aligned} J_h(\theta_2) &= \frac{1}{2}\sum_{t=1}^{T} \frac{1}{\frac{1}{e_{1,k+1}^2(t)} + \frac{1}{e_{2,k}^2(t)} + \cdots + \frac{1}{e_{N,k}^2(t)}} \\ &= \frac{1}{2}\sum_{t=1}^{T} \underbrace{\frac{\frac{1}{e_{2,k}^2(t)}}{\frac{1}{e_{1,k+1}^2(t)} + \frac{1}{e_{2,k}^2(t)} + \cdots + \frac{1}{e_{N,k}^2(t)}}}_{w_{2,k}^2(t)} e_{2,k}^2(t) \\ &= \frac{1}{2}\sum_{t=1}^{T} w_{2,k}^2(t)(y(t+1) - \varphi^T(t)\theta_2)^2 \\ &= \frac{1}{2}\sum_{t=1}^{T} (w_{2,k}(t)y(t+1) - w_{2,k}(t)\varphi^T(t)\theta_2)^2 \end{aligned} \tag{2.44}$$

where, again, the identification error of the first model has been re-calculated as

$$e_{1,k+1}(t) = y(t+1) - \varphi^T(t)\theta_1(k+1) \tag{2.45}$$

The optimal solution can be computed as

$$\theta_2^* = (\Phi_{2,k}^T \Phi_{2,k})^{-1} \Phi_{2,k}^T Y_{2,k} \tag{2.46}$$

where

$$Y_{2,k} = (w_{2,k}(1)y(2), w_{2,k}(2)y(3), \ldots, w_{2,k}(T)y(T+1))^T \tag{2.47}$$

$$\Phi_{2,k} = \begin{pmatrix} w_{2,k}(1)\varphi^T(1) \\ \vdots \\ w_{2,k}(T)\varphi^T(T) \end{pmatrix} \tag{2.48}$$

Thus let

$$\theta_2(k+1) = (\Phi_{2,k}^T \Phi_{2,k})^{-1} \Phi_{2,k}^T Y_{2,k} \tag{2.49}$$

Similarly, the m^{th} step is to update the parameter of the m^{th} model as follows:

$$\theta_m(k+1) = (\Phi_{m,k}^T \Phi_{m,k})^{-1} \Phi_{m,k}^T Y_{m,k} \tag{2.50}$$

where

$$Y_{m,k} = (w_{m,k}(1)y(2), w_{m,k}(2)y(3), \ldots, w_{m,k}(T)y(T+1))^T \tag{2.51}$$

$$\Phi_{m,k} = \begin{pmatrix} w_{m,k}(1)\varphi^T(1) \\ \vdots \\ w_{m,k}(T)\varphi^T(T) \end{pmatrix} \tag{2.52}$$

$$w_{m,k}^2(t) = \frac{\dfrac{1}{e_{m,k}^2(t)}}{\dfrac{1}{e_{1,k+1}^2(t)} + \cdots + \dfrac{1}{e_{m-1,k+1}^2(t)} + \dfrac{1}{e_{m,k(t)}^2} + \cdots + \dfrac{1}{e_{N,k}^2(t)}} \tag{2.53}$$

As mentioned, the convergence property for this cost function is much harder to be analyzed. Nevertheless, extensive simulation studies do show that it decreases to zero when data is sufficiently rich. Another advantage of minimizing this cost function is that it provides parameter estimates which are less sensitive to measurement noise due to its normalization property, i.e. the weights are always normalized to be between 0 and 1.

Remark 2.2 *In the previous sections, the number of the models, N, is assumed known. However, if it were unknown a priori, how do we choose a suitable N?*

For noiseless data, having any number larger than or equals the true N will provide the same fitting accuracy. This is because at any one time, we would still have one of the identification errors $e_m(i)$ being zero, rendering both the geometrical mean and harmonic mean of squares of errors to be zero. It is easy to then discard the redundant models by observing their never-zero identification errors.

As for noisy data, one possible guidance to gauge the suitability of a particular N is the fitting accuracy of the models on both training and test data. For the training data, it is intuitive that the larger N is, the more accurate the data fitting will be. However, this might not necessarily be reflected on the test set because of overfitting. Therefore, we could try out a few different N's and choose the one which gives the best balance between the wellness-of-fit and that of the test data.

Remark 2.3 *Another question is how to define the order of the multiple local models. We will show later via simulation in chapter 3.6.1 that overestimating the order of the models will not affect the accuracy of the models much.*

2.4.3 Simulation Study of Noiseless Case

Consider the following switched system with four second order subsystems

$$y(t+1) = \begin{cases} -0.5y(t) - 0.9y(t-1) + 0.8u(t) + 0.3u(t-1) & \text{if subsystem 1} \\ -0.7y(t) - 0.3y(t-1) + 0.2u(t) + 0.4u(t-1) & \text{if subsystem 2} \\ -0.3y(t) - 0.2y(t-1) + 0.6u(t) + 0.2u(t-1) & \text{if subsystem 3} \\ -0.6y(t) - 0.8y(t-1) + 0.2u(t) + 0.1u(t-1) & \text{if subsystem 4} \end{cases} \tag{2.54}$$

The initial system outputs $y(0)$ and $y(-1)$ as well as $u(-1)$ are set as zero. We generated 100 data points $y(1) \dots y(100)$. The control input $u(0) \dots u(99)$ is generated randomly in the range of [0, 10]. The subsystem which is active at a particular instant is also chosen randomly with equal probability.

The least geometrical mean squares algorithm provides us the correct parameter estimates in less than ten iterations, as shown in TABLE 2.1. Thus, the efficacy of the proposed algorithm is validated.

Table 2.1: Parameter Estimates of a Switched System with Four Subsystems

Iterations	a11	a12	b11	b12	a21	a22	b21	b22
0 (randomly generated)	0.6174	0.7427	0.9960	0.9279	0.0191	0.3812	0.6570	0.0282
1	-0.5601	-0.8792	0.6691	0.3658	-0.3383	-0.1449	0.4115	0.1450
2	-0.6115	-0.9577	0.7194	0.3202	-0.5507	-0.1536	0.2398	0.3144
3	-0.5696	-0.9034	0.8054	0.2707	-0.6306	-0.1849	0.1895	0.3713
4	-0.5070	-0.8997	0.8050	0.2902	-0.6707	-0.2658	0.1928	0.3878
5	-0.5002	-0.9000	0.7999	0.3001	-0.7005	-0.2998	0.1998	0.4004
6	-0.5000	-0.9000	0.8000	0.3000	-0.7000	-0.3000	0.2000	0.4000
7	-0.5000	-0.9000	0.8000	0.3000	-0.7000	-0.3000	0.2000	0.4000
Iterations	a31	a32	b31	b32	a41	a42	b41	b42
0 (randomly generated)	0.0462	0.8809	0.4224	0.4557	0.8323	0.1927	0.6799	0.9183
1	-0.7039	-0.5999	0.2697	0.6450	-0.3715	-0.4770	0.3032	-0.0487
2	-0.6921	-0.5443	0.4168	0.7487	-0.5647	-0.7619	0.2363	0.0647
3	-0.5690	-0.4615	0.4964	0.6048	-0.5876	-0.7897	0.2064	0.0917
4	-0.4095	-0.3016	0.5447	0.3771	-0.5976	-0.8010	0.2027	0.0989
5	-0.3015	-0.2020	0.5983	0.2034	-0.5979	-0.7990	0.2002	0.0991
6	-0.3000	-0.2000	0.6000	0.2000	-0.6000	-0.8000	0.2000	0.1000
7	-0.3000	-0.2000	0.6000	0.2000	-0.6000	-0.8000	0.2000	0.1000

Table 2.2: Parameter Estimates of the Four Subsystems, Noisy Case

Iterations	a11	a12	b11	b12	a21	a22	b21	b22
0 (randomly generated)	0.8319	0.7627	0.7633	0.7014	0.5459	0.4076	0.2125	0.6275
1	-0.5443	-0.7739	0.6156	0.3139	-0.6553	-0.6086	0.2142	0.4529
2	-0.4039	-0.7023	0.7190	0.2528	-0.7472	-0.5928	0.2931	0.5371
3	-0.3722	-0.7166	0.7073	0.3074	-0.8160	-0.4638	0.2498	0.5358
4	-0.4535	-0.8576	0.7492	0.3214	-0.7956	-0.3292	0.2188	0.4680
5	-0.4734	-0.8943	0.7848	0.2993	-0.7097	-0.3050	0.2170	0.3874
6	-0.4766	-0.8915	0.7866	0.2997	-0.7033	-0.3105	0.2106	0.3924
7	-0.4773	-0.8920	0.7869	0.2998	-0.7033	-0.3109	0.2102	0.3930
8	-0.4773	-0.8921	0.7869	0.2998	-0.7032	-0.3110	0.2102	0.3931
9	-0.4773	-0.8921	0.7869	0.2998	-0.7032	-0.3110	0.2101	0.3931
10	-0.4773	-0.8921	0.7869	0.2998	-0.7032	-0.3110	0.2101	0.3931
Iterations	a31	a32	b31	b32	a41	a42	b41	b42
0 (randomly generated)	0.8180	0.1878	0.1969	0.2116	0.1679	0.6145	0.6366	0.7694
1	0.5344	-0.1364	0.4730	-0.1052	0.7550	-0.5158	0.2484	-0.7022
2	0.1756	-0.1280	0.4886	0.0098	-0.6774	-0.6913	0.2105	0.1260
3	-0.1379	-0.1118	0.5249	0.0259	-0.5994	-0.8021	0.2094	0.0916
4	-0.1843	-0.1373	0.5773	0.0151	-0.6029	-0.7939	0.2079	0.0902
5	-0.3104	-0.1779	0.5992	0.1628	-0.5994	-0.7918	0.2065	0.0885
6	-0.3191	-0.1791	0.6002	0.1686	-0.5971	-0.7891	0.2056	0.0872
7	-0.3198	-0.1792	0.6003	0.1691	-0.5969	-0.7887	0.2055	0.0871
8	-0.3199	-0.1792	0.6003	0.1691	-0.5969	-0.7887	0.2055	0.0871
9	-0.3199	-0.1792	0.6003	0.1691	-0.5969	-0.7887	0.2055	0.0871
10	-0.3199	-0.1792	0.6003	0.1691	-0.5969	-0.7887	0.2055	0.0871

2.4.4 Simulation Study of Noisy Case

We consider the same system in section 2.4.3, with the output corrupted with a zero-mean noise with standard deviation 0.1. The proposed algorithm can again provide good estimates of the system parameters in this noisy case, as shown in TABLE 2.2. This shows that the algorithm is robust in face of measurement noise.

2.4.5 Comparison Studies

As pointed out in the introduction, there indeed exist some other methods for the identification of switched systems, for instance the algebraic-geometric approach proposed in [77] and its extensions. In this section, we would like to show the advantages of our algorithm, as compared to the mentioned algebraic-geometric approach.

We consider again the same system in section 2.4.3. The algorithm in [77] relies on transforming the system equations into one single equation called the "hybrid decoupling constraint (HDC)", identifying the parameters of this HDC, and then recovering back the identified HDC parameters into the parameters of each subsystem. It was reported that there are $\binom{n+n_a+n_b}{n}$ independent monomials in the HDC, where n is the total number of subsystems, n_a and n_b are the numbers of coefficients involving $y(t), y(t-1)\ldots$ and $u(t), u(t-1)\ldots$ respectively in each subsystem. This means that for the system considered with $n = 4$, $n_a = 2$ and $n_b = 2$, the HDC is a 70^{th}-order polynomial, with 69 parameters to be identified, even though there are only 16 parameters to be identified in the original switched system. This "sudden" increase of the parameters to be identified is one weak point of the algorithm, especially when we have a switched system with many subsystems, or when each of the subsystems is of higher order.

Secondly, the algorithm in [77] is also much more vulnerable to noise than our proposed algorithms. In our simulation studies, we first add to system in section 2.4.3 a zero-mean white noise with standard deviations of 0.1. We then run each algorithm ten times, and calculate the mean of the ten collected values. The result is shown in TABLE 2.3. As can be seen from TABLE 2.3, our least geometric mean (LGM) and least harmonic mean (LHM) algorithms provide better estimates of the system parameters.

Next, we repeat the same simulations, this time with a white noise of standard devia-

Table 2.3: Mean of the Identified Parameters, Noise $N(0, 0.01)$

Parameter	True value	LGM	LHM	[77]
a11	-0.5	-0.4949	-0.4941	-0.4792
a12	-0.9	-0.9024	-0.8989	-0.8395
b11	0.8	0.8027	0.8023	0.7703
b12	0.3	0.2900	0.2938	0.2934
a21	-0.7	-0.6981	-0.6910	-0.6413
a22	-0.3	-0.3075	-0.3080	-0.2600
b21	0.2	0.2043	0.2032	0.1729
b22	0.4	0.3950	0.3860	0.3708
a31	-0.3	-0.2688	-0.3042	-0.3875
a32	-0.2	-0.2011	-0.2163	-0.2221
b31	0.6	0.6070	0.6070	0.6359
b32	0.2	0.1682	0.1914	0.1834
a41	-0.6	-0.5896	-0.5937	-0.5991
a42	-0.8	-0.8025	-0.7980	-0.8158
b41	0.2	0.2013	0.2016	0.1972
b42	0.1	0.0990	0.0943	0.0906

tion 0.2 added to the system. For this level of noise, we observe that sometimes the least geometric mean algorithm and, more severely, the algorithm of [77] give identification results which are very far away from the true values. We shall ignore these results which are unacceptably wrong, and collect only ten reasonable results. We then calculate the mean of these ten collected values. The result is shown in TABLE 2.4.

As can be seen from TABLE 2.4, our least geometrical mean (LGM) and least harmonic mean (LHM) algorithms not only provide better estimates of the system parameters, but are also much more robust to noise than the algorithm in [77]. The robustness is inferred from the smaller number of attempts needed to obtain the ten acceptable identification results. Also, we see that the least harmonic mean algorithm is the least susceptible to noise. This is because the weights used in the least harmonic mean algorithm are normalized to be less than or equal to one.

Finally, the algorithm given by [77] is not readily extendible to nonlinear systems, whereas it can be seen later that our algorithm can be utilized for the identification of nonlinear systems as well.

2.5 Online Identification of Linear Time-Varying Systems

2.5.1 The Gradient Descent Algorithm

Here, we use the gradient descent algorithm to identify the parameters of the switched linear system online. At the i^{th} step, the cost function to minimize is

$$J_g(\theta_1, \ldots, \theta_m, \ldots, \theta_N) = \frac{1}{2} e_1^2(i) \ldots e_m^2(i) \ldots e_N^2(i) \tag{2.55}$$

which is a special case of (2.8). Again, by fixing

$$w_m(i) = \prod_{\substack{j=1 \\ j \neq m}}^{N} e_j(i) \tag{2.56}$$

Table 2.4: Mean of the Identified Parameters, Noise $N(0, 0.04)$

Parameter	True value	LGM	LHM	[77]
a11	-0.5	-0.5298	-0.4868	-0.4925
a12	-0.9	-0.9054	-0.8955	-0.8774
b11	0.8	0.8120	0.8051	0.8075
b12	0.3	0.2961	0.2837	0.3186
a21	-0.7	-0.6554	-0.7009	-0.6459
a22	-0.3	-0.3276	-0.3218	-0.2491
b21	0.2	0.1794	0.1969	0.2343
b22	0.4	0.3748	0.4010	0.3524
a31	-0.3	-0.3116	-0.2715	-0.3503
a32	-0.2	-0.2030	-0.2129	-0.2455
b31	0.6	0.6106	0.6022	0.5963
b32	0.2	0.1881	0.1942	0.2265
a41	-0.6	-0.6065	-0.5878	-0.6119
a42	-0.8	-0.7900	-0.8007	-0.7953
b41	0.2	0.1994	0.1943	0.2300
b42	0.1	0.0971	0.1022	0.0959
Total runs needed to get ten reasonable results		15	10	31
Failed attempts		5	0	21

we obtain

$$\begin{aligned} J_g(\theta_m) &= \frac{1}{2} w_m^2(i) e_m^2(i) \\ &= \frac{1}{2} w_m^2(i)(y(i+1) - \varphi^T(x(i))\theta_m(i))^2 \end{aligned} \tag{2.57}$$

The gradient of the cost function at time t is

$$\begin{aligned} \frac{\partial J_g(\theta_m)}{\partial \theta_m} &= -w_m^2(t)(y(t+1) - \varphi^T(x(t))\theta_m(t))\varphi(x(t)) \\ &= -w_m^2(t) e_m(t)\varphi(x(t)) \end{aligned} \tag{2.58}$$

Thus, the updated θ_m estimate is calculated as

$$\theta_{m,t+1} = \theta_{m,t} + \eta w_m^2(t) e_m(t)\varphi(x(t)) \tag{2.59}$$

To ensure convergence of the gradient descent algorithm, the "weighted learning rate" ηw_m^2 is chosen as follows:

$$\eta w_m^2(t) = \frac{1}{1 + \varphi^T(x(t))\varphi(x(t))} \cdot \frac{w_m^2}{\sum_{j=1}^{N} w_j^2} \tag{2.60}$$

Remark 2.4 *The conventional projection algorithm*

$$\theta(t+1) = \theta(t) + \frac{1}{c + \varphi^T(t)\varphi(t)} a e(t)\varphi(t) \tag{2.61}$$

is guaranteed to converge if $0 < a < 2$ *and* $c > 0$*, a result which has been proven using Lyapunov analysis. Thus, the convergence of our gradient descent algorithm (2.59) with the weighted learning rate (2.60) is also ensured, since* $0 < a = \frac{w_m^2}{\sum_{j=1}^{N} w_j^2} < 1 < 2$ *and* $c = 1 > 0$.

Remark 2.5 *The gradient descent algorithm (2.59) with the weighted learning rate (2.60) is the same as that proposed in [19]. This shows that the algorithm can be derived from our proposed point of view.*

2.5.2 Simulation Study of Noiseless Case

We consider the same system in section 2.4.3. The simulation results for the online parameter estimation using the proposed gradient descent algorithm are shown in Fig. 2.1.

As can be seen from Fig. 2.1, the subsystem parameters can be identified correctly using the proposed algorithm.

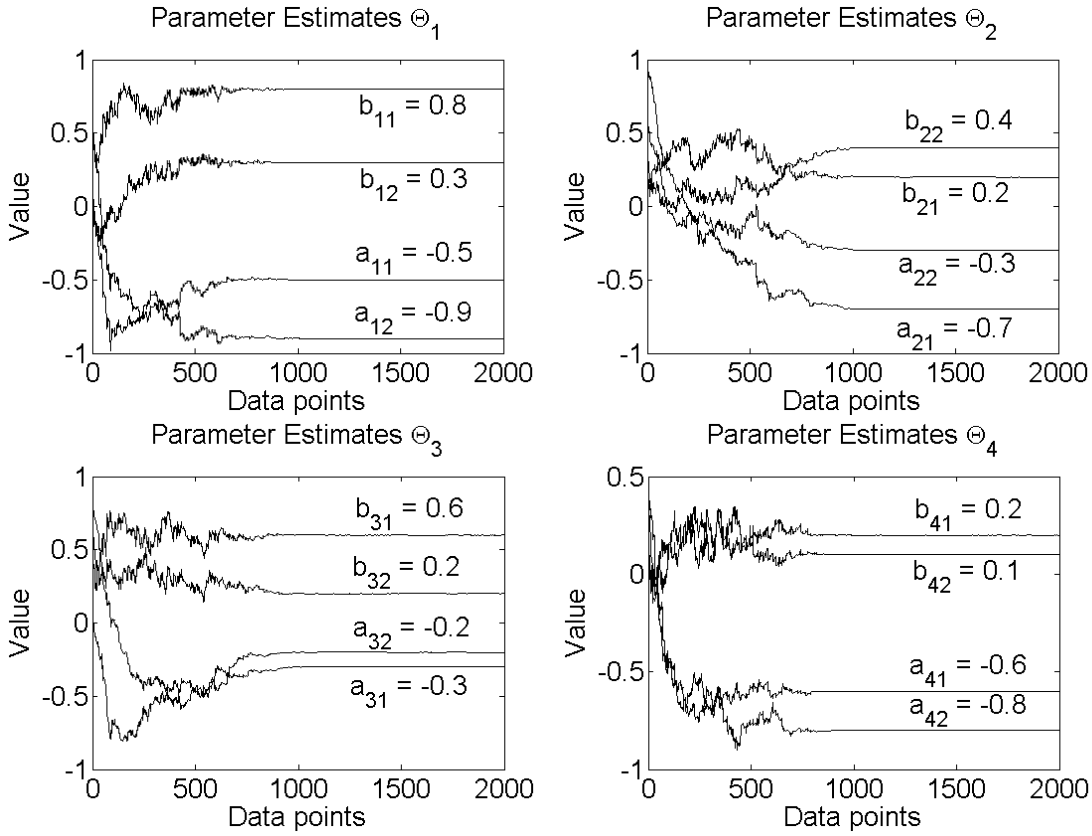

Figure 2.1: Parameter estimates using the gradient descent algorithm

2.5.3 Simulation Study of Noisy Case

Again, the same system is being considered, with the output corrupted with a zero-mean noise with standard deviation 0.1. The evolution of the parameter estimates is shown in Fig. 2.2.

It can be clearly seen that despite the measurement noise, the system parameters can still be estimated using the proposed algorithm.

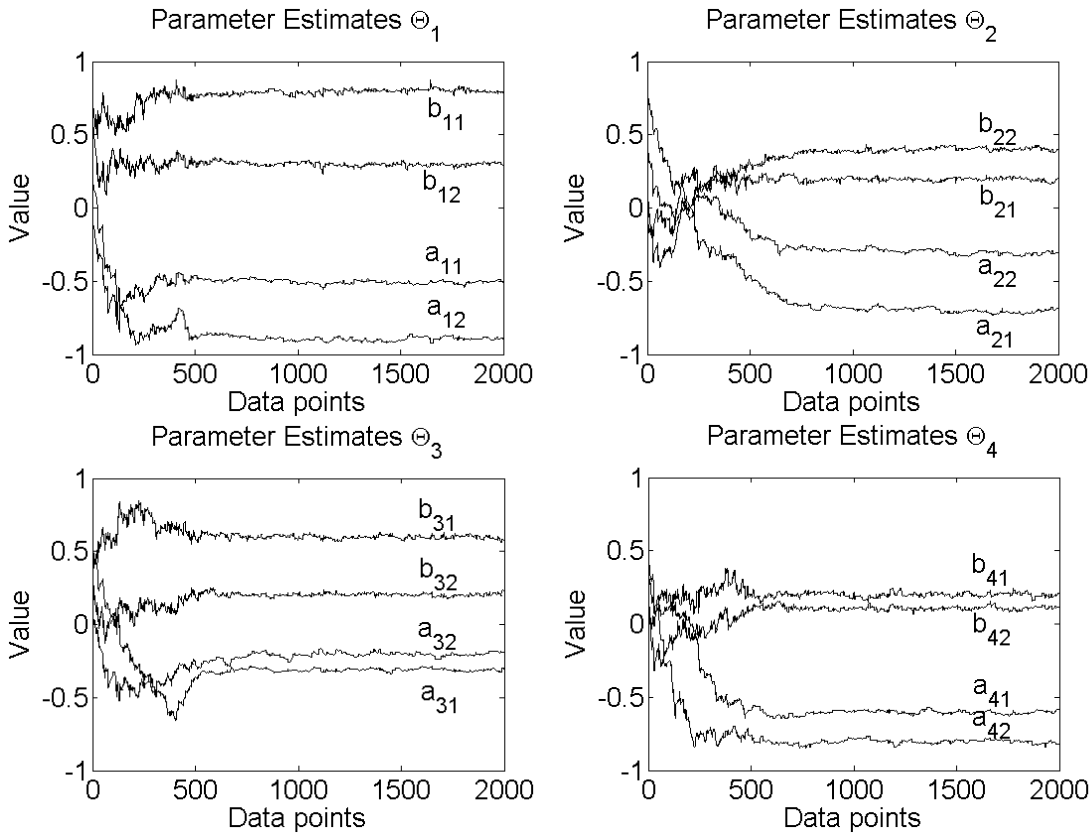

Figure 2.2: Parameter estimates using the gradient descent algorithm, noisy case

2.6 Identification of Time-Varying Nonlinear Systems

Now, let's consider the identification of time-varying nonlinear systems. Assume that the system can be approximated as a switching system with nonlinear subsystems described by equation (2.6). The equation is repeated here for an easier readability.

$$y(t+1) = \begin{cases} f_1(x(t), \theta_1^*) & \text{if subsystem } 1 \\ \vdots & \vdots \\ f_m(x(t), \theta_m^*) & \text{if subsystem } m \in (1, \ldots, N) \\ \vdots & \vdots \\ f_N(x(t), \theta_N^*) & \text{if subsystem } N \end{cases} \tag{2.62}$$

Our aim in this section is to obtain an accurate estimation of the nonlinear functions $f_m(x(t), \theta_m)$, $m = \{1, \ldots, N\}$. Two popular tools for modeling nonlinear systems are the radial basis function network (RBFN) and the multilayer perceptron (MLP), which are both well-known as universal approximators. The RBFN is actually linear in the parameters, and thus all the previously developed algorithms can be applied to obtain its parameters/weights. Therefore, we shall concentrate on the MLP in this section.

More specifically, we will use N MLP's to approximate the N subsystems individually.

$$y(t+1) = \begin{cases} f_1(x(t), \theta_1^*) & \text{if subsystem } 1 \\ \vdots & \vdots \\ f_m(x(t), \theta_m^*) & \text{if subsystem } m \\ \vdots & \vdots \\ f_N(x(t), \theta_N^*) & \text{if subsystem } N \end{cases} \approx \begin{cases} \text{MLP}_1 & \text{if subsystem } 1 \\ \vdots & \vdots \\ \text{MLP}_m & \text{if subsystem } m \\ \vdots & \vdots \\ \text{MLP}_N & \text{if subsystem } N \end{cases} \tag{2.63}$$

The next question is how to train the N MLP's so that each of them approximates one subsystem of the original system.

2.6.1 The 'Weighted Back Propagation' Algorithm

The standard way of training the parameters of a single multilayer perceptron is the back propagation method. Here, we propose the 'weighted back propagation' algorithm to train the N MLP's simultaneously for modeling the N nonlinear subsystems of a switched system.

The 'weighted back propagation' algorithm is derived by minimizing the cost function (2.55) by using the gradient descent algorithm. As mentioned in our general framework in section 2.3.2, we shall fix all other errors from other MLP's as the weight for updating the parameters of the m^{th} MLP. Thus, (2.55) can be viewed as:

$$J_g(\theta_m) = \frac{1}{2} w_m^2(i) e_m^2(i) = \frac{1}{2} w_m^2(i)(y(i+1) - \hat{y}_m(i+1))^2 \tag{2.64}$$

where

$$w_m(i) = \prod_{\substack{j=1 \\ j \neq m}}^{N} e_j(i) \tag{2.65}$$

Without going into the details of the derivations, which we defer to the Appendix A, we would like to point out that the parameters of the MLP's are updated in almost the same way as in the usual back propagation algorithm. The only modification necessary is to weigh the local gradient of the output layer as

$$\delta_1^{(output)} = w_m^2(y(i+1) - \hat{y}_m(i+1)) \tag{2.66}$$

or

$$\delta_1^{(output)} = \frac{w_m^2}{\sum_{j=1}^{N} w_j^2}(y(i+1) - \hat{y}_m(i+1)) \tag{2.67}$$

where we prefer the latter for its robustness against noise, because the weight is normalized to be between zero and one, although the learning rate becomes slightly slower.

2.6.2 Simulation Study of Noiseless Case

Consider the following switched system with three underlying nonlinear subsystems

$$y(t+1) = \begin{cases} (3 + \sin(2y(t)))u(t) & \text{if subsystem 1} \\ 0.1y(t) + \dfrac{u(t)(2 + u^2(t))}{1 + 2u^2(t)} & \text{if subsystem 2} \\ y(t) - 0.1y^3(t) - u(t) & \text{if subsystem 3} \end{cases} \tag{2.68}$$

A set of 1000 data $y(1) \ldots y(1000)$ is generated, with $y(0)$ set as zero and $u(0) \ldots u(999)$ randomly generated in the range of [0, 1]. The subsystem which is active at a particular instant is also chosen randomly with equal probability.

The MLP's we use have two input nodes (one for $u(t)$ and $y(t)$ each), one hidden layer with nine nodes, and one output node (for $\hat{y}_m(t+1)$). The activation function for the hidden layer is the sigmoidal function, whereas the activation function for the output layer is identity.

The number of epochs is set as 20000, while the learning rate is chosen as 0.1. After training the three MLP's, we generated for each subsystem a new set of data with random input $u(0) \ldots u(39)$ in the range of [0, 1], and tested out how well the MLP's approximate the three subsystems. The simulation result is shown in Fig. 3.5.

As can be seen from Fig. 3.5, each of the three MLP's indeed specialized for one of the subsystems, and the function approximation is also quite satisfactory. The small residue could be further decreased by running more epochs.

2.6.3 Simulation Study of Noisy Case

Consider again the same system as described in section 2.6.2. This time however, the training data is corrupted with a zero-mean white noise with standard deviation 0.2. After training the three MLP's using the noisy data, we generated for each subsystem a new set of noiseless data with random input $u(0) \ldots u(39)$ in the range of [0, 1], and tested out how well the MLP's approximate the three subsystems. The simulation result is shown in Fig. 2.4.

As can be observed from Fig. 2.4, the proposed algorithm allows accurate approximation of the three nonlinear subsystems, even when the training data is noisy. This verifies the robustness of the algorithm.

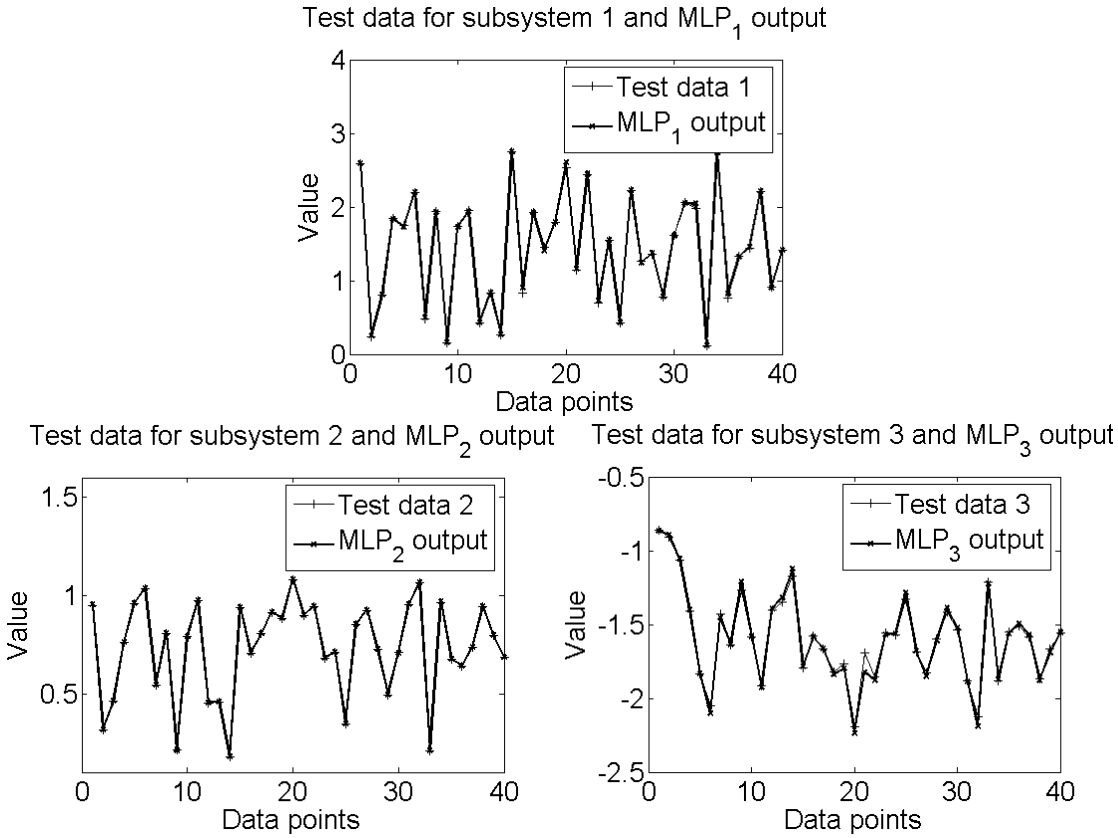

Figure 2.3: Test data vs. the output of the three MLP's

2.7 Conclusions

In this chapter, we presented a general framework for the identification of time varying system using multiple models. The time varying discrete-time system is approximated by a switched system. The underlying subsystems of the switched system can be linear or nonlinear. For the linear case, the models are the usual linear regression model, whereas for the nonlinear case, multilayer perceptrons are used to perform function approximation. We showed that by minimizing appropriate cost functions, i.e. the geometric mean squares or the harmonic mean squares, we can identify the subsytems with sufficient accuracy. Extensive simulation results verified the efficacy of our proposed algorithm. We also obtained a preliminary convergence proof of the parameter estimates

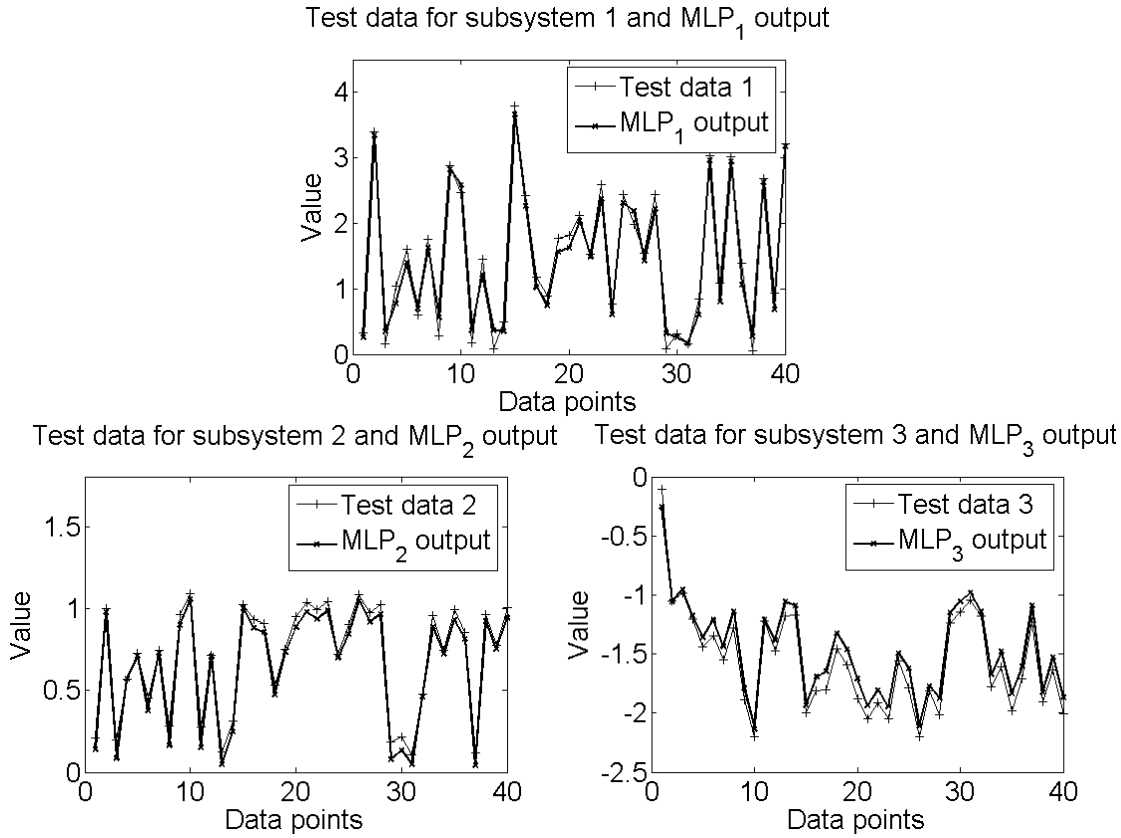

Figure 2.4: Test data vs. the output of the three MLP's, noisy case

for the case with linear subsystems, which we will further develop in the future.

Chapter 3

Identification of Piecewise Affine Systems and Nonlinear Systems using Multiple Models

3.1 Introduction

Piecewise affine (PWA) systems are systems whose state-input domain is partitioned into a finite number of non-overlapping regions, with each individual subsystem in the different regions being linear or affine [30, 60, 72]. If the subsystem in each region has an ARX (AutoRegressive systems with eXogenous inputs) input-output relationship, then the system is termed Piecewise affine ARX (PWARX) system [30, 60]. PWA systems received a lot of attention from researchers because they are equivalent to several classes of hybrid models [26] and thus can be used to obtain hybrid models from data, whereby typical examples of hybrid systems include manufacturing systems, telecommunication networks, traffic control systems, digital circuits, and logistic systems [17]. Moreover, PWA models can be used to approximate nonlinear dynamical systems by switching among various linear/affine models [60].

The identification of PWARX models is very challenging, because both the param-

eters of the affine subsystems and the partitions of the regressor domain have to be estimated [60]. The main difficulty is that the parameter identification problem is coupled with a data classification problem, whereby each data point needs to be associated with the most suitable submodel [60].

Some methods to identify PWA and PWARX models have been proposed in the literature [6, 20, 24, 28, 29, 50, 67, 77]. In [24], the system consisting of different modes or subsystems is viewed as a Hidden Markov Model, and an Expectation-Maximization (EM) algorithm, or more precisely, the Baum-Welch algorithm, is used to obtain the parameters of the subsystems as well as the parameters of the transition probability matrix. The algebraic geometric approach in [77] transforms the multiple ARX models into a single, "lifted" ARX model that does not depend on the switching sequence. The parameters of the "lifted" ARX model is identified using standard linear identification techniques, and finally the parameters of the original ARX submodels are recovered. While the algebraic geometric approach provides a closed-form solution to the identification in the absence of noise, it has been observed that the algorithm is rather sensitive to noise or nonlinear disturbances [30]. [20, 28, 50] proposed using clustering techniques to collect the input-output data into different groups, and then estimate the parameter vector for each cluster. However, in the cases when the model orders are not known exactly, the identification results become very poor, since distances in the feature space become corrupted by irrelevant information. [67] formulated the identification of two subclasses of PWA models as mixed-integer linear or quadratic problems, and solved them using available optimization algorithms. The cost functions are guaranteed to converge to the global optimum, but the worst-case complexity is very high and the procedure is therefore only suitable for cases where relatively few data are available. In [29], a Bayesian

procedure was derived to estimate the parameter vectors, which are treated as random variables and described through their probability density functions. The Bayesian procedure works well if sufficient physical insight into the underlying data generating process is available, but poor initialization may lead to poor identification results. The bounded-error procedure in [6] fits a PWARX model satisfying $|e(t)| = |y_{t+1} - f(r_t)| \leq \epsilon$, without any assumption on the system generating the data. This procedure is well suited for the cases when there is no *a priori* knowledge on the system, and when one wishes to identify a model with prescribed bounded prediction error. However, it may be difficult to find the right combination of tuning parameters. The readers are also referred to [30] and [60] which provide a summary and comparison of some of the procedures mentioned above.

In this chapter, we propose a procedure for the identification of PWARX systems, which can also be applied to identify nonlinear systems by approximating them via PWARX models. The proposed algorithm is not only simple but also accurate. Extensive simulation studies show that our algorithm can indeed provide accurate estimates of the plant parameters even in noisy cases, and even when the model orders are overestimated. The method is then used to obtain a model of an electric motor that exhibits PWA characteristics due to output saturation, as well as a model of a single-link robotic arm.

This chapter is organized as follows. In section 3.2, the problem formulation is given. The two steps of the proposed identification algorithm are then detailed in section 3.3 and section 3.4, followed by a discussion on approximating nonlinear systems using PWARX models in section 3.5. Simulation results are shown in section 3.6, whereas experimental results on the modeling of the electric motor system with PWA characteristics and single-link robotic arm are shown in section 3.7. Finally, in section 3.8, conclusions will be

drawn.

3.2 Problem Formulation

Consider a PWARX system with N partitions such that

$$y(t+1) = \begin{cases} \varphi^T(t)\theta_1^* & \text{if } x(t) \in X_1 \\ \vdots & \vdots \\ \varphi^T(t)\theta_m^* & \text{if } x(t) \in X_{m\in(1,\ldots,N)} \\ \vdots & \vdots \\ \varphi^T(t)\theta_N^* & \text{if } x(t) \in X_N \end{cases} \tag{3.1}$$

where

$$x(t) = [y(t), \ldots, y(t-n_a), u(t), \ldots, u(t-n_b)]^T \tag{3.2}$$

$$\varphi(t) = \left[x^T(t), 1\right]^T \tag{3.3}$$

and

$$\cup_{i=1}^{N} X_i = X \text{ and } X_i \cap X_j = \emptyset, \forall i \neq j \tag{3.4}$$

where X denotes the whole regressor space.

Our aim is to identify the parameters of each subsystem, namely $\theta_1^*, \ldots, \theta_N^*$ as well as to estimate the regions $X_1, \ldots, X_N$.

Secondly, we also aim to approximate a nonlinear function

$$\begin{aligned} \xi(t+1) &= f(\xi(t), u(t)) \\ y(t) &= h(\xi(t)) \end{aligned} \tag{3.5}$$

where the output $y(t) \in R$, the states $\xi(t) \in R^n$ and the input $u(t) \in R$ are discrete time sequences, via PWARX models given in equation (3.1).

As mentioned in the introduction, the identification of PWA systems is very challenging, because the parameter identification is coupled with a data classification problem, whereby each data point needs to be associated with the most suitable submodel. If

the partitions of the regressor space is known *a priori*, then the problem can be easily solved, as the data can be easily separated into different groups (submodels), and the parameters of each individual submodel can be identified in a straightforward manner using standard linear identification methods. However, if the partitions are unknown, how can we identify the PWA systems? This is the problem we are going to address in this chapter.

3.3 First Step: Parameter Identification

In the first step of our algorithm, the parameters of the subsystems are identified. The partitions of the regressor domain are estimated only in the second step.

It can be observed that the PWARX system (3.1) is a special case of the general switching linear system (2.7), which is repeated here for the convenience of the readers.

$$y(t+1) = \begin{cases} \varphi^T(x(t))\theta_1^* & \text{if subsystem } 1 \\ \vdots & \vdots \\ \varphi^T(x(t))\theta_m^* & \text{if subsystem } m \in (1,\ldots,N) \\ \vdots & \vdots \\ \varphi^T(x(t))\theta_N^* & \text{if subsystem } N \end{cases} \tag{3.6}$$

The two specialties of PWARX systems are that firstly, the underlying subsystems have input-output relationships of the affine ARX form, i.e.

$$\varphi(x(t)) \rightarrow \varphi(t) = \left[x^T(t), 1\right]^T \tag{3.7}$$

$$x(t) = [y(t),\ldots,y(t-n_a),u(t),\ldots,u(t-n_b)]^T \tag{3.8}$$

and secondly, the switching signal is dependent on the location of the regression vector $x(t)$ within the regressor space X.

As such, the Least Geometrical Mean and Least Harmonic Mean algorithms described in 2.4 and 2.5 can be easily adopted to cater for parameter identification of the

subsystems, since these two algorithms could solve the parameter identification problem without any concern on how the subsystems switch from one to another. Therefore, we will not repeat the development of the parameter identification algorithms in this chapter.

3.4 Second Step: Estimation of the Partition of the Regressor Space

After the parameters of the individual subsystems have been identified by using the proposed Least Geometrical Mean and Least Harmonic Mean algorithms, the next step is to estimate the the partitioned regions $X_1, \ldots, X_N$. This will allow us to determine the active subsystem via the location of the regression vector $x(t) = [y(t), \ldots, y(t-n_a), u(t), \ldots, u(t-n_b)]^T$ within the regressor space X, and in turn makes it possible for the next output to be predicted based on the current active subsystem.

This can be formulated as a standard pattern classification problem, and the steps are briefly detailed as follows: The (training) data are first labelled based on the minimum prediction error from the multiple models, i.e. class (i^{th}data point) = arg $\min_{1\leq j\leq N}(|e_j(i)|)$, and grouped into different classes. Standard algorithms such as multi-class support vector machine classifiers and neural network classifiers are then trained (with inputs=$x(t)$ and output=class) to classify the data and to estimate the boundaries of the partitions. As a notation, we will write

$$\sigma(t) = h(x(t)) \tag{3.9}$$

where h is the function of the data classifier which maps the input vector $x(t)$ to $\sigma(t)$, with $\sigma(t) = \{1, \ldots, N\}$ denoting the active subsystem.

3.4.1 Standard Regressor Space - Classifier I

According to the standard definition of the PWARX system (3.1), the regression vector is $x(t) = [y(t), \ldots, y(t-n_a),\ \ u(t), \ldots, u(t-n_b)]^T$. $x(t)$ can therefore be directly used as the input of the data classifier (3.9). Hereafter, the classifier in this case will be called "Classifier I" (see Fig 3.1(a)).

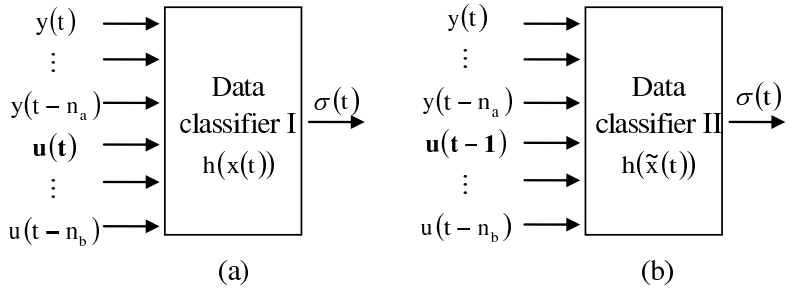

Figure 3.1: Data classifier for estimation of partition of regressor space

3.4.2 Modified Regressor Space - Classifier II

However, for reasons which will be explained later, including $u(t)$ within the regression vector $x(t)$ in Classifier I will prohibit an efficient use of the predictive control. We therefore propose to omit $u(t)$ from the regression vector, and use the "modified regressor" $\tilde{x}(t) = [y(t), \ldots, y(t-n_a), u(t-1), \ldots, u(t-n_b)]^T$ as the input to the data classifier (3.9). This classifier is named "Classifier II" (see Fig 3.1(b)).

As will be seen later in the simulation studies, for certain nonlinear systems, this modification is acceptable for the identification purpose and would not degrade the accuracy of the PWA model severely. It is however noteworthy that the performance of model-based predictive control can be improved significantly by using this modified regressor for data classification.

Remark 3.1 *To avoid confusion, we would like to stress here that the modified regression*

vector applies only for the data classification step, i.e. we omit $u(t)$ only when classifying the data points. The full regressor is still being used for the parameter identification part. That means, equation (3.1) is rewritten as

$$y(t+1) \approx \begin{cases} \varphi^T(t)\theta_1^* & \text{if } \tilde{x}(t) \in \tilde{X}_1 \\ \vdots & \vdots \\ \varphi^T(t)\theta_m^* & \text{if } \tilde{x}(t) \in \tilde{X}_{m\in(1,\ldots,N)} \\ \vdots & \vdots \\ \varphi^T(t)\theta_N^* & \text{if } \tilde{x}(t) \in \tilde{X}_N \end{cases} \tag{3.10}$$

where

$$x(t) = [y(t), \ldots, y(t-n_a), u(t), \ldots, u(t-n_b)]^T \tag{3.11}$$

$$\varphi(t) = \left[x^T(t), 1\right]^T \tag{3.12}$$

$$\tilde{x}(t) = [y(t), \ldots, y(t-n_a), u(t-1), \ldots, u(t-n_b)]^T \tag{3.13}$$

and

$$\cup_{i=1}^{N} \tilde{X}_i = \tilde{X} \text{ and } \tilde{X}_i \cap \tilde{X}_j = \emptyset, \forall i \neq j \tag{3.14}$$

3.5 Nonlinear Systems Approximation

Consider a nonlinear system

$$\begin{aligned} \xi(t+1) &= f(\xi(t), u(t)) \\ y(t) &= h(\xi(t)) \end{aligned} \tag{3.15}$$

where the output $y(t) \in R$, the states $\xi(t) \in R^n$ and the input $u(t) \in R$ are discrete time sequences. Assume that system (3.15) can also be described by the following nonlinear Auto-Regressive Moving Average (NARMA) model:

$$y(t+1) = F(y(t), \ldots, y(t-n_a), u(t), \ldots, u(t-n_b)) \tag{3.16}$$

The reader can refer to [81] for detailed discussion on the conditions of the existence of such global input-output models for nonlinear systems.

As mentioned earlier, given a set of input-output data of the above nonlinear system, PWA models can be used to approximate $F(\cdot)$ by switching among various linear/affine models

$$y(t+1) \approx \begin{cases} \varphi^T(t)\theta_1^* & \text{if } x(t) \in X_1 \\ \vdots & \vdots \\ \varphi^T(t)\theta_m^* & \text{if } x(t) \in X_{m\in(1,\ldots,N)} \\ \vdots & \vdots \\ \varphi^T(t)\theta_N^* & \text{if } x(t) \in X_N \end{cases} \tag{3.17}$$

Therefore, our proposed algorithm can be directly used for identifying the nonlinear systems. The accuracy of the approximation depends on the choice of number of piecewise affine subsystems, as well as the dimension of the regression vector (n_a, n_b).

3.6 Simulation Studies

3.6.1 Piecewise Affine Systems 1

The example given in this section is based on a simulation study in [30]. We use this example for better comparison of our algorithm with those given in the work.

The PWARX system is as follows:

$$y(t+1) = \begin{cases} 2y(t) + 10 + \eta(t) & \text{if } y(t) \in [-10, 0) \\ -1.5y(t) + 10 + \eta(t) & \text{if } y(t) \in [0, 10] \end{cases} \tag{3.18}$$

The accuracy of the identified parameters is measured through the quantity

$$\Delta_\theta = \underbrace{\max}_{1\le i\le N} \left(\underbrace{\min}_{1\le j\le N} \frac{\|\theta_i - \theta_j^*\|_2}{\|\theta_j^*\|_2} \right) \tag{3.19}$$

For perfect estimates, Δ_θ is zero. However, it increases as the estimates worsen.

The overall quality measure which also includes the estimation of the active region

can be provided by the sum of squared residuals (one step ahead prediction errors):

$$\hat{\sigma}_\epsilon^2 = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathrm{SSR}_{Fi}}{|Fi|} \tag{3.20}$$

where the set Fi contains the data points classified to the submodel i, $|Fi|$ denotes the cardinality or the number of elements in the set Fi, and the sum of squared residuals (SSR) of model i is defined as:

$$\mathrm{SSR}_{Fi} = \sum_{x(t)\in Fi}(y(t+1) - \varphi^T(t)\theta_i)^2 \tag{3.21}$$

This value not only measures the accuracy of the subsystem parameters, but also how well the correct partition is chosen based on the regression vector. The value is considered acceptable if $\hat{\sigma}_\epsilon^2$ is small and/or near the expected noise of the identified system.

First of all, we are going to test our algorithm for both noiseless and noisy data, when the order of the system is known.

For noiseless data, the least geometrical mean squares (LGM) algorithm and least harmonic mean squares (LHM) algorithm provide us the correct parameter estimates in just a few iterations, as shown in TABLE 3.1 and TABLE 3.2 respectively. Thus, the efficacy of the proposed algorithm is validated.

We now study the effects of noise $\eta(t)$ onto the identification algorithm. $\eta(t)$ is normally distributed with zero mean and increasing variance σ_n^2. The simulation design is as follows. The data set consists of 100 noisy data points, based on which our algorithm should identify the parameters of the subsystems. For each noise level, we repeat the simulation ten times and calculate the mean of Δ_θ. The simulation result is shown in TABLE 3.3. The result of using the LHM algorithm is comparable to that of using the Bayesian procedure, the clustering-based procedure and the bounded-error procedure as shown in [30].

Table 3.1: Parameter Estimates of the PWARX System 1 using Least Geometrical Mean Squares Algorithm

Iterations	a1	c1
0 (randomly generated)	0.2379	0.6458
1	0.8461	4.0008
2	1.3524	6.5457
3	1.9027	9.4943
4	2.0000	10.000
5	2.0000	10.000
Iterations	a2	c2
0 (randomly generated)	0.1603	0.8729
1	-1.0501	7.3440
2	-1.1828	7.3801
3	-1.3921	9.1214
4	-1.4989	9.9909
5	-1.5000	10.000

For the readers to have a feel of the values of Δ_θ in TABLE 3.3, two examples are that the parameter estimates of $(1.7814, 8.8215)$ and $(-1.4614, 9.8198)$ would give $\Delta_\theta = 0.1175$, whereas the parameter estimates of $(1.9814, 9.8734)$ and $(-1.4955, 9.9567)$ would give $\Delta_\theta = 0.0125$. From these results, we see that even in noisy cases, our algorithms, especially the least harmonic mean squares algorithm, can still give accurate estimates of the parameters.

Our next simulation studies are to test the effects of overestimating the model orders. For these simulations, we set the noise level at $\sigma_n^2 = 0.01$. The true regressor for the system is $[y(t), 1]$ with $n_a = 0$ and n_b irrelevant, but we use the following regressors $[y(t), \ldots, y(t-n_a), u(t), \ldots, u(t-n_b), 1]$ for all combinations of $n_a = 0, \ldots, 3$ and $n_b = 0, \ldots, 4$ to model the true system, where $u(t)$ was generated as random number between -10 and 10. If our algorithms work well for these cases, we expect the parameters corresponding to the redundant elements in the regressor to be identified as approxi-

Table 3.2: Parameter Estimates of the PWARX System 1 using Least Harmonic Mean Squares Algorithm

Iterations	a1	c1
0 (randomly generated)	0.3663	0.4435
1	-0.0393	3.5572
2	0.1744	3.8749
3	0.4035	4.2033
...	...	...
12	2.0000	10.000
13	2.0000	10.000
14	2.0000	10.000
Iterations	a2	c2
0 (randomly generated)	0.3663	0.4435
1	-0.5944	4.3440
2	-1.0036	6.7468
3	-1.2546	8.3988
...	...	...
12	-1.4995	9.9965
13	-1.5000	10.000
14	-1.5000	10.000

Table 3.3: Means of Δ_θ for several noise level σ_n^2

σ_n^2	0.01	0.02	0.04	0.06
LGM	0.0078	0.0135	0.0301	0.0374
LHM	0.0034	0.0067	0.0086	0.0104
σ_n^2	0.08	0.10	0.12	0.14
LGM	0.0607	0.0695	0.0779	0.1145
LHM	0.0193	0.0196	0.0275	0.0279

mately zero - which is indeed true, as shown by the simulation result for the extreme case of $n_a = 3$ and $n_b = 4$ in TABLE 3.4

Figure 3.2 and figure 3.3 show the values of $log_{10}(\sigma_\epsilon^2)$ for models with different model orders by using the LGM and LHM algorithms respectively. Recall that the noise level

Table 3.4: Parameter estimation for overestimated model order

parameter	true	estimate	parameter	true	estimate
a_{11}	2	2.0003	a_{21}	-1.5	-1.4954
a_{12}	0	0.0216	a_{22}	0	0.0004
a_{13}	0	-0.0251	a_{23}	0	0.0008
a_{14}	0	-0.0021	a_{24}	0	0.0004
b_{11}	0	0.0003	b_{21}	0	-0.0016
b_{12}	0	0.0003	b_{22}	0	0.0003
b_{13}	0	0.0000	b_{23}	0	-0.0007
b_{14}	0	-0.0002	b_{24}	0	0.0007
b_{15}	0	-0.0002	b_{25}	0	-0.0003
bias_1	10	9.8551	bias_2	10	9.9719

is 0.01, thus a value of $log_{10}(\sigma_\epsilon^2) = -2$ is considered satisfactory. We see that the parameter estimates by LGM algorithm is affected slightly for overestimated n_a but is not affected by n_b, and the accuracy level is considered acceptable. On the other hand, the LHM algorithm gives accurate parameter estimates regardless of the assumed model order. This shows that our algorithms have little or no difficulties in estimating the over-parameterized model. This is an advantage of our algorithms over most of the other algorithms given in [30].

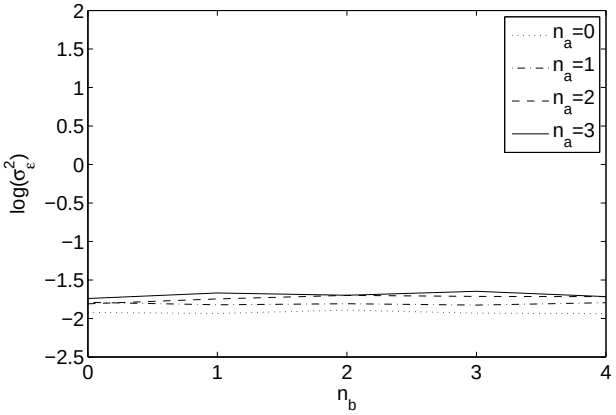

Figure 3.2: σ_ϵ^2 using LGM algorithm

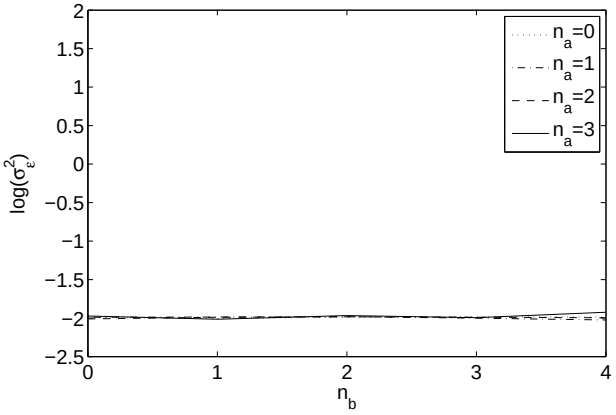

Figure 3.3: σ_ϵ^2 using LHM algorithm

3.6.2 Piecewise Affine Systems 2

Consider the following PWARX system:

$$y(t+1) = \begin{cases} -a_{13}y(t) - a_{12}y(t-1) - a_{11}y(t-2) \\ -a_{10}y(t-3) + b_{13}u(t) + b_{12}u(t-1) \qquad \text{if } x(t) \in X_1 \\ +b_{11}u(t-2) + b_{10}u(t-3) + c_{10} \\ \\ -a_{23}y(t) - a_{22}y(t-1) - a_{21}y(t-2) \\ -a_{20}y(t-3) + b_{23}u(t) + b_{22}u(t-1) \qquad \text{if } x(t) \in X_2 \\ +b_{21}u(t-2) + b_{20}u(t-3) + c_{20} \end{cases} \tag{3.22}$$

where $x(t) = [y(t), \ldots, y(t-3), u(t), \ldots, u(t-3)]^T$. The plant parameters are $a_{13} = 0.2$, $a_{12} = 0.8$, $a_{11} = 0.7$, $a_{10} = 0.1$, $b_{13} = 0.3$, $b_{12} = 0.5$, $b_{11} = 0.6$, $b_{10} = 0.3$, $c_{10} = 1.2$, $a_{23} = 0.6$, $a_{22} = 0.4$, $a_{21} = 0.9$, $a_{20} = 0.3$, $b_{23} = 0.7$, $b_{22} = 0.05$, $b_{21} = 0.2$, $b_{20} = 0.4$, and $c_{20} = 1.7$. The region X_1 is active if $h^T x(t) < 0$, whereas the region X_2 is active if $h^T x(t) \geq 0$, with $h = 0.2\,[1\ 1\ 1\ 1\ 1\ 1\ 1\ 1]^T$.

The system, with zero initial values, is being excited using random control input $u(t)$ in the range of $[-13, 13]$. A total of 4000 input-output data are obtained. The parameters of the two input-output models in (3.22) are then identified using the least geometrical mean algorithm as detailed in section 2.4.1, and the two regions X_1 and X_2

are estimated using multilayer perceptrons.

To test the accuracy of the identified PWARX model, we excite (3.22) with the following control signal:

$$u(t) = 7(\sin(\pi t/10) + \sin(\pi t/20)) \tag{3.23}$$

and check whether the identified PWARX model can give a good prediction of the output. The test result is shown in Fig. 3.4. From the fact that the true output (dashed line) is almost completely covered by the estimated output (solid line), it can be concluded that the identified PWARX model is very accurate.

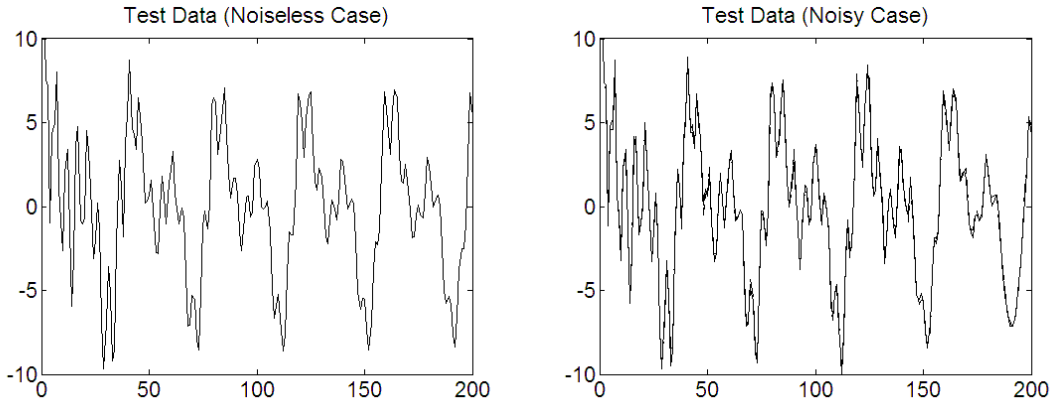

Figure 3.4: Output prediction for PWA system 2 using the identified PWARX model.
Dashed: True output, solid: Estimated output.
Left: Noiseless case, right: Noisy case

3.6.3 Nonlinear Systems 1

Consider the following nonlinear system,

$$y(t+1) = \frac{y(t)}{1+y^2(t)} + u^3(t) \tag{3.24}$$

which can be found in [53].

The training set consists of 4000 data, with the inputs exciting the system being random inputs in the range of $[-2, 2]$.

The PWARX model is of the form

$$y(t+1) \approx \begin{cases} \varphi^T(t)\theta_1^* & \text{if } x(t) \in X_1 \\ \vdots & \vdots \\ \varphi^T(t)\theta_N^* & \text{if } x(t) \in X_N \end{cases} \tag{3.25}$$

$$\text{where} \qquad x(t) = [y(t), u(t)]^T$$

$$\varphi(t) = \left[x(t)^T, 1\right]^T$$

After identifying the parameters θ_1^* to θ_N^* using either the Least Geometrical Mean algorithm or Least Harmonic Mean algorithm, we proceed to train a neural network for data classification using the regressor $x(t)$.

To verify the accuracy of the PWARX model in approximating the original nonlinear system, the output of the nonlinear system is compared with the output of the PWARX model when the test input is described by

$$u(t) = \sin(2\pi t/10) + \sin(2\pi t/25) \tag{3.26}$$

The simulation result for the test data is shown in Fig. 3.5. As can be observed, the nonlinear system can be approximated well using the PWARX model with 4 or 16 subsystems.

To have a more quantitative evaluation of the accuracy of the identified model, we compute the "Fit"-value which is defined as

$$\text{Fit} = \left(1 - \sqrt{\frac{\sum_t (y(t+1) - \hat{y}(t+1))^2}{\sum_t (y(t+1) - \bar{y})^2}}\right) \times 100\% \tag{3.27}$$

where $y(t+1)$, $\hat{y}(t+1)$ and $\bar{y}$ are the true output, estimated output, and the average of $y(t+1)$ respectively. The result is shown in TABLE 3.5. The values show that the data-fitting improves as the number of subsystems increases.

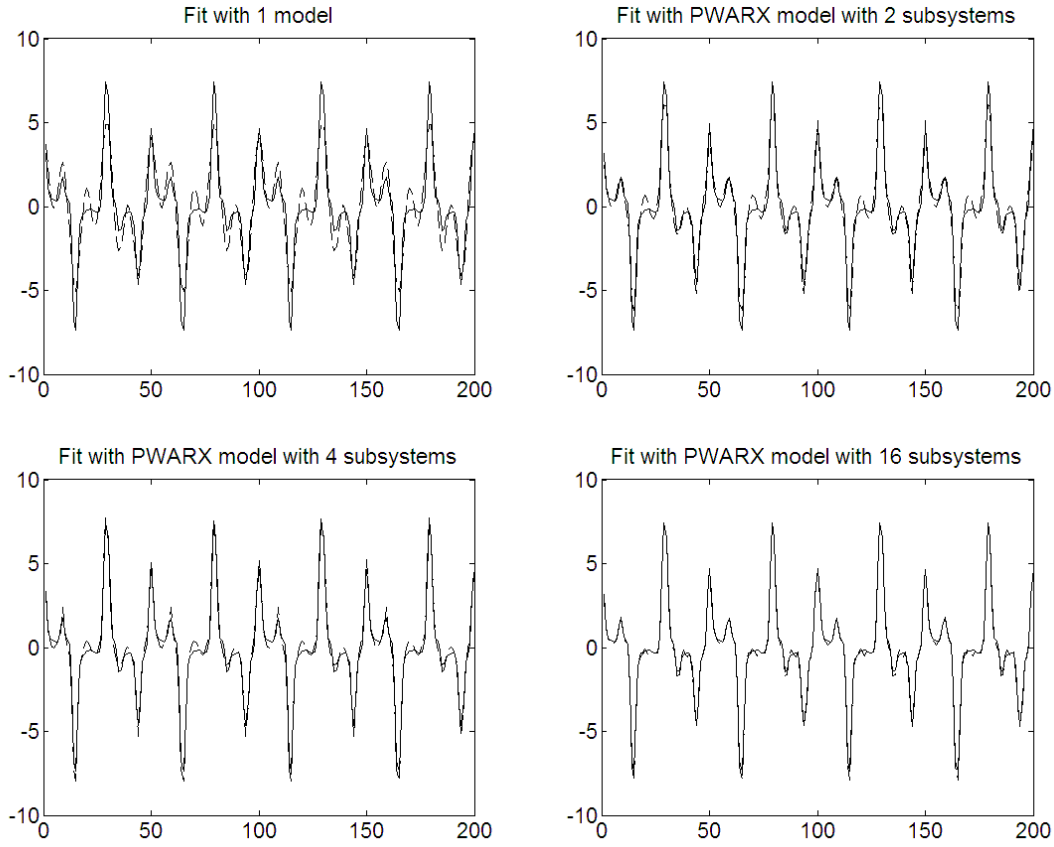

Figure 3.5: Identification of nonlinear system 1 via PWARX models.
Solid: True output, dashed: Estimated output.

3.6.4 Nonlinear Systems 2

Consider the following nonlinear system,

$$\begin{aligned} y(t+1) = & \frac{1.5y(t)y(t-1)}{1+y^2(t)+y^2(t-1)} \\ & + \sin(y(t)+y(t-1)) + u(t) + 0.8u(t-1) \end{aligned} \tag{3.28}$$

This system is a combination of the examples by [13] and [53], with the parameters slightly modified to make the nonlinearity even more profound.

Table 3.5: Fit values for nonlinear system 1

No. subsystems	1	2	4	8	16
Fit Values	60%	77%	85%	91%	93%

The training set consists of 4000 data, with the inputs exciting the system being random inputs in the range of $[-2.2, 2.2]$.

The PWARX model is of the form

$$y(t+1) \approx \begin{cases} \varphi^T(t)\theta_1^* & \text{if } x(t) \in X_1 \text{ or } \tilde{x}(t) \in \tilde{X}_1 \\ \vdots & \vdots \\ \varphi^T(t)\theta_N^* & \text{if } x(t) \in X_N \text{ or } \tilde{x}(t) \in \tilde{X}_N \end{cases} \tag{3.29}$$

where

$$x(t) = [y(t), y(t-1), u(t), u(t-1)]^T \tag{3.30}$$

$$\varphi(t) = \left[x(t)^T, 1\right]^T \tag{3.31}$$

$$\tilde{x}(t) = [y(t), y(t-1), u(t-1)]^T \tag{3.32}$$

After identifying the parameters θ_1^* to θ_N^* using either the Least Geometrical Mean algorithm or Least Harmonic Mean algorithm, we proceed to train a neural network for data classification, using either the standard regressor $x(t)$ or the modified regressor $\tilde{x}(t)$.

To verify the accuracy of the PWARX model in approximating the original nonlinear system, the output of the nonlinear system is compared with the output of the PWARX model when the test input is described by

$$u(t) = \sin(2\pi t/10) + \sin(2\pi t/25) \tag{3.33}$$

The simulation results for the test data by using Classifier I and Classifier II to choose the active subsystem are shown in Figs. 3.6 and 3.7 respecively. As can be observed, the nonlinear system can be approximated well using the PWARX model with 4 or 8 subsystems. Also, we see that the fitting accuracy is not degraded much by omitting $u(t)$ from the data classifier.

The quantitative accuracy of the PWARX models are computed using the "Fit"-value

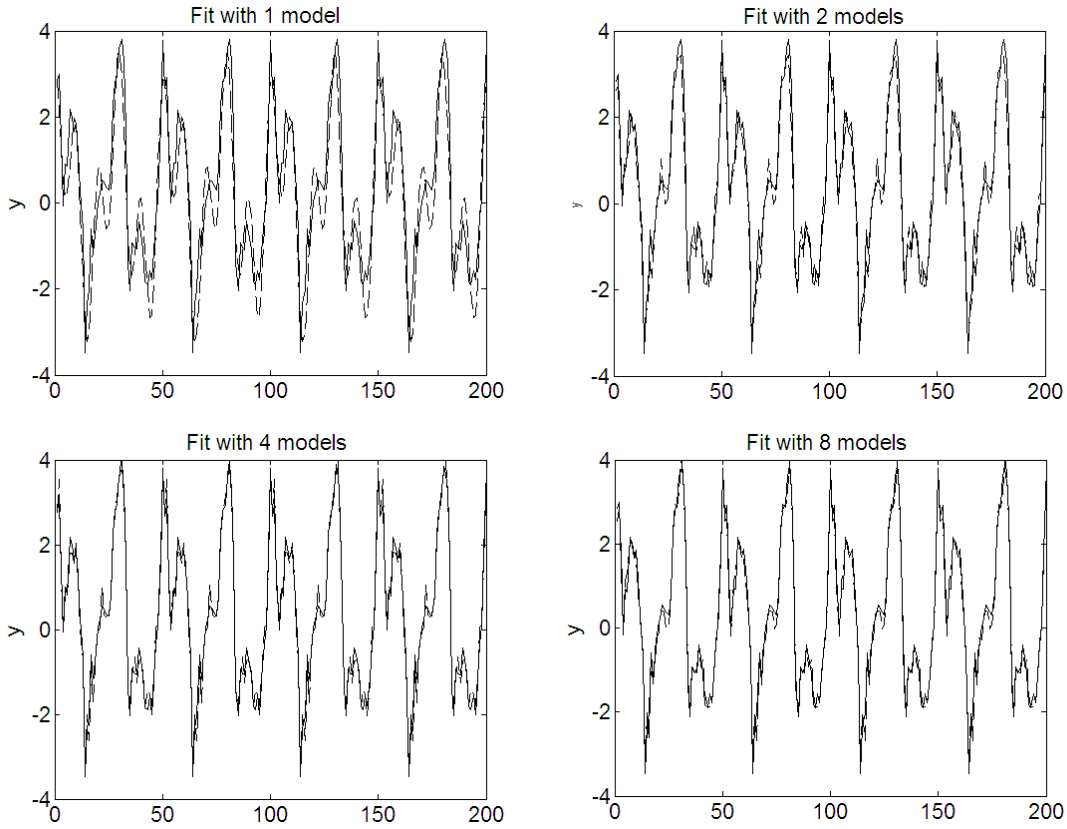

Figure 3.6: Identification of the nonlinear system 2 via PWARX model using Classifier I. Solid: True output, dashed: Estimated output.

which is defined as

$$\text{Fit} = \left(1 - \sqrt{\frac{\sum_t (y(t+1) - \hat{y}(t+1))^2}{\sum_t (y(t+1) - \bar{y})^2}}\right) \times 100\% \tag{3.34}$$

for both cases where either Classifier I or II is used to decide the active subsystem, where $y(t+1)$, $\hat{y}(t+1)$ and $\bar{y}$ are the true output, estimated output, and the average of $y(t+1)$ respectively. The results, given in TABLE 3.6, show that the model accuracy are very similar for both types of classifier.

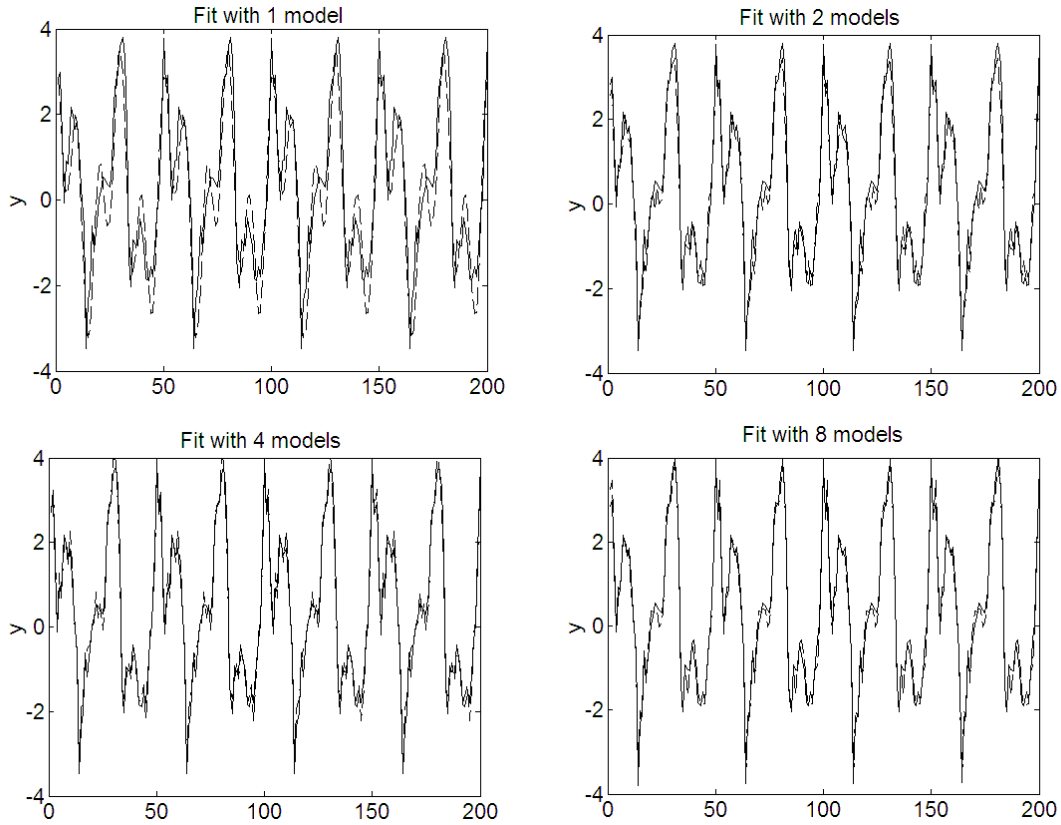

Figure 3.7: Identification of the nonlinear system 2 via PWARX model using Classifier II. Solid: True output, dashed: Estimated output.

3.6.5 Nonlinear Systems 3

Consider a nonlinear benchmark system given in the state space form:

$$\begin{aligned}
\xi_1(t+1) &= \left(\frac{\xi_1(t)}{1+\xi_1^2(t)} + 1\right)\sin(\xi_2(t)) \\
\xi_2(t+1) &= \xi_2(t)\cos(\xi_2(t)) + \xi_1(t)\mathrm{e}^{-(\xi_1^2(t)+\xi_2^2(t))/8} \\
&\quad + \frac{u^3(t)}{1+u^2(t)+0.5\cos(\xi_1(t)+\xi_2(t))} \\
y(t) &= \frac{\xi_1(t)}{1+0.5\sin(\xi_2(t))} + \frac{\xi_2(t)}{1+0.5\sin(\xi_1(t))} + \epsilon(t)
\end{aligned} \tag{3.35}$$

Table 3.6: Fit values for nonlinear system 2

No. subsystems	1	2	4	8
Classifier I	59.1%	80.2%	85.9%	87.7%
Classifier II	59.1%	79.9%	85.3%	86.7%

This plant, taken from [55], does not represent any real system, but is sufficiently complex and nonlinear so that the conventional linear methods will not provide satisfactory performance. This example was also used in [68] and [79] to test the efficacy of direct weight optimization (DWO) and piecewise-affine-basis models (BPWA) in approximating nonlinear systems.

The training set consists of 4000 data, with the inputs exciting the system being random inputs in the range of $[-2.5, 2.5]$.

The PWARX model is of the form

$$y(t+1) \approx \begin{cases} \varphi^T(t)\theta_1^* & \text{if } x(t) \in X_1 \text{ or } \tilde{x}(t) \in \tilde{X}_1 \\ \vdots & \vdots \\ \varphi^T(t)\theta_N^* & \text{if } x(t) \in X_N \text{ or } \tilde{x}(t) \in \tilde{X}_N \end{cases} \tag{3.36}$$

where

$$\begin{aligned} x(t) &= [y(t), y(t-1), y(t-2), u(t), u(t-1), u(t-2)]^T \\ \varphi(t) &= \left[x(t)^T, 1\right]^T \\ \tilde{x}(t) &= [y(t), y(t-1), y(t-2), u(t-1), u(t-2)]^T \end{aligned} \tag{3.37}$$

After identifying the parameters θ_1^* to θ_N^* using either the Least Geometrical Mean algorithm or Least Harmonic Mean algorithm, we proceed to train a neural network for data classification, using either the standard regressor $x(t)$ or the modified regressor $\tilde{x}(t)$.

To verify the accuracy of the PWARX model in approximating the original nonlinear system, the output of the nonlinear system is compared with the output of the PWARX model when the test input is described by

$$u(t) = \sin(2\pi t/10) + \sin(2\pi t/25) \tag{3.38}$$

The simulation results for the test data by using Classifier I and Classifier II to choose the active subsystem are shown in Figs. 3.8 and 3.9 respectively. As can be observed, the

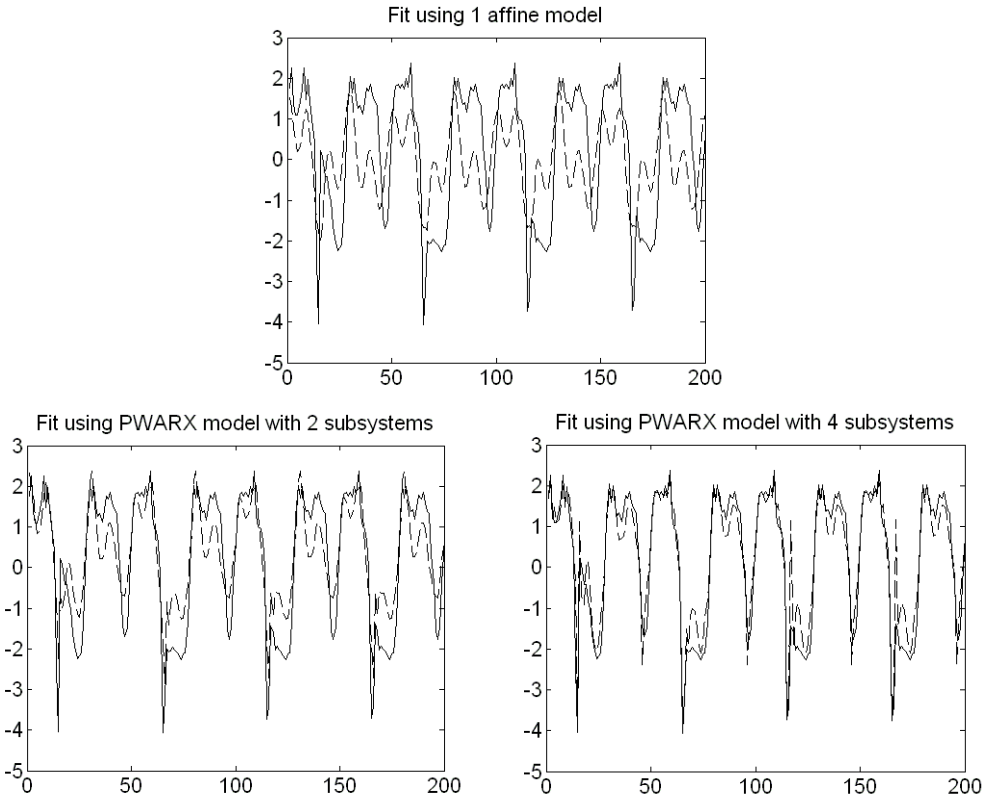

Figure 3.8: Identification of the nonlinear system 3 via PWARX model - Classifier I. Solid: True output, dashed: Estimated output.

nonlinear system can be approximated well using the PWARX model with 4 subsystems. Also, we see that the fitting accuracy is not degraded much by omitting $u(t)$ from the data classifier.

The quantitative accuracy of the PWARX models are computed using the "Fit"-value which is defined as

$$\text{Fit} = \left(1 - \sqrt{\frac{\sum_t (y(t) - \hat{y}(t))^2}{\sum_t (y(t) - \bar{y})^2}}\right) \times 100\% \tag{3.39}$$

for both cases where either Classifier I or II is used to decide the active subsystem, where $y(t)$, $\hat{y}(t)$ and $\bar{y}$ are the true output, estimated output, and the average of $y(t)$ respectively. The result, given in TABLE 3.7, shows that the model accuracy are rather similar for both types of classifier.

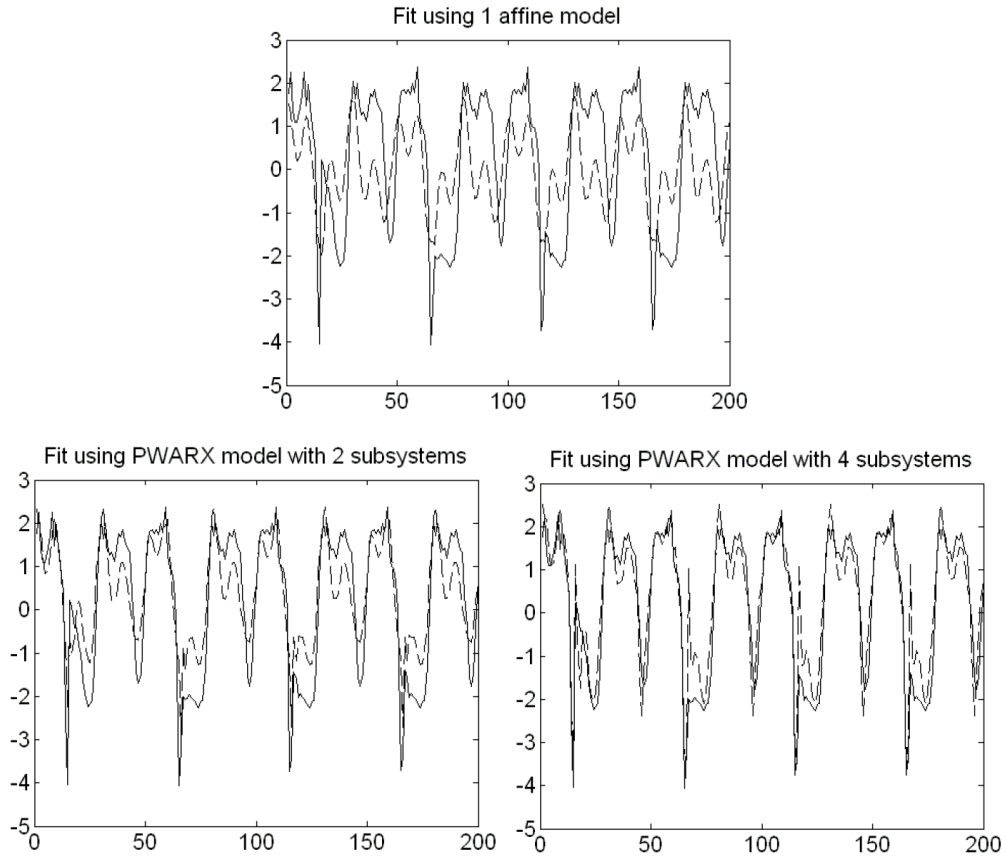

Figure 3.9: Identification of the nonlinear system 3 via PWARX model - Classifier II. Solid: True output, dashed: Estimated output.

3.7 Experimental Studies

3.7.1 Electric Motor Systems with Velocity Saturation

The experimental testbed for the case study is an electronically commutated electric motor, a Maxon EC-powermax30 (200W) [46] (Fig. 3.10). The reduction ratio of the mounted gear-head is 113:1 and its power efficiency is 80%. An encoder with 500 pulses/turn is used to measure the motor side angle, from which the angular velocity can then be calculated.

In the provided specification [46], the maximum angular velocity of the motor is given as 17000 RPM (i.e. approximately 15.7 rad/s at the end of the gear head). This velocity

Table 3.7: Fit values for nonlinear example 3

No. subsystems	1	2	4
Classifier I	16.5%	50.8%	68.4%
Classifier II	16.5%	48.7%	62.2%

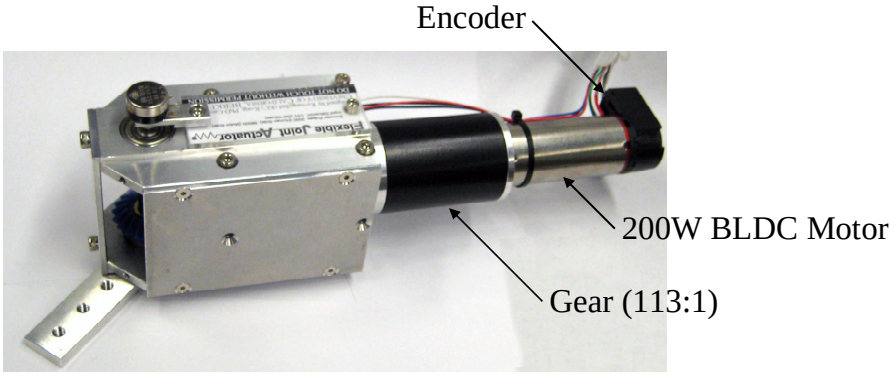

Figure 3.10: The geared motor system used as experimental testbed

saturation is caused by the maximum voltage provided to the motor windings, as well as the limitations of the servo amplifier output [35].

The velocity saturation can be observed from several measurements. For example, in Fig. 3.11 where the motor is fed with step inputs of varying magnitudes, it is noticed that before the maximum velocity is reached, the steady state velocity is almost proportional to the magnitude of the step input, but once the velocity reaches the saturation value, an increase in the control input does not affect the steady state velocity of the motor any further.

PWARX Model of the Motor System

Because of the velocity saturation, the electric motor may be viewed as a PWA system, with one of the subsystems representing the case when the motor is operating within the linear region at lower speed, and the other subsystem representing the case

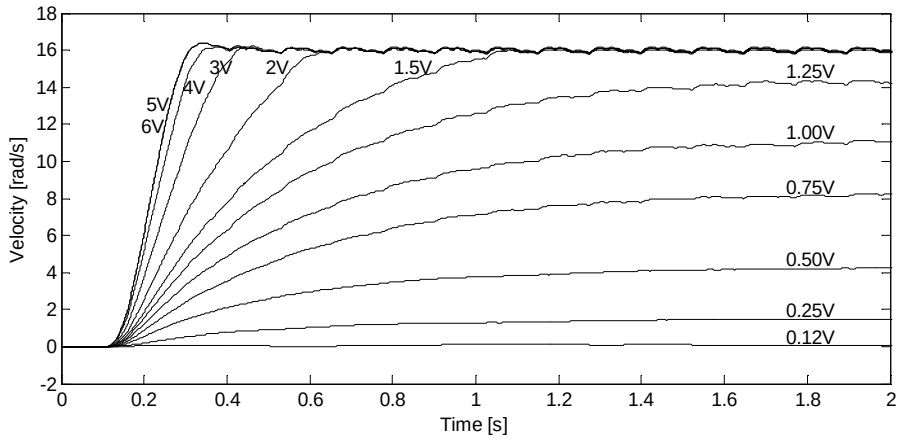

Figure 3.11: Velocity responses to step inputs with different magnitudes

when the velocity saturation is reached. Thus, it serves as an appropriate testbed for the application of the proposed PWARX model identification algorithm.

Identification for the Training Set

First of all, the input signal is designed to be a chirp-like signal, with the amplitude being 5V, and the frequencies of the sine waves being 0.1Hz, 0.2Hz, 0.5Hz, 1Hz, 2Hz, 5Hz, 10Hz and 20Hz, one after another. We are interested in these lower frequencies because the motor/gear system does not reach the velocity saturation speed when excited by inputs with higher frequences [35]. The input and output signals are then sampled at 50Hz. This set of data is used to train the PWARX model which is in the form of:

$$\hat{v}(t+1) = \begin{cases} \theta_{11}v(t) + \theta_{12}u(t) + \theta_{13} & \text{if } [v(t), u(t)] \in X_1 \\ \theta_{21}v(t) + \theta_{22}u(t) + \theta_{23} & \text{if } [v(t), u(t)] \in X_2 \end{cases} \tag{3.40}$$

The parameters of the two subsystems are identified using the proposed least geometrical mean squares (LGM) algorithm and the least harmonic mean squares (LHM) algorithm, whereas the two regions X_1 and X_2 are estimated using a 2-12-2 multilayer perceptrons.

To showcase the advantage of the PWARX model, the following linear model is also

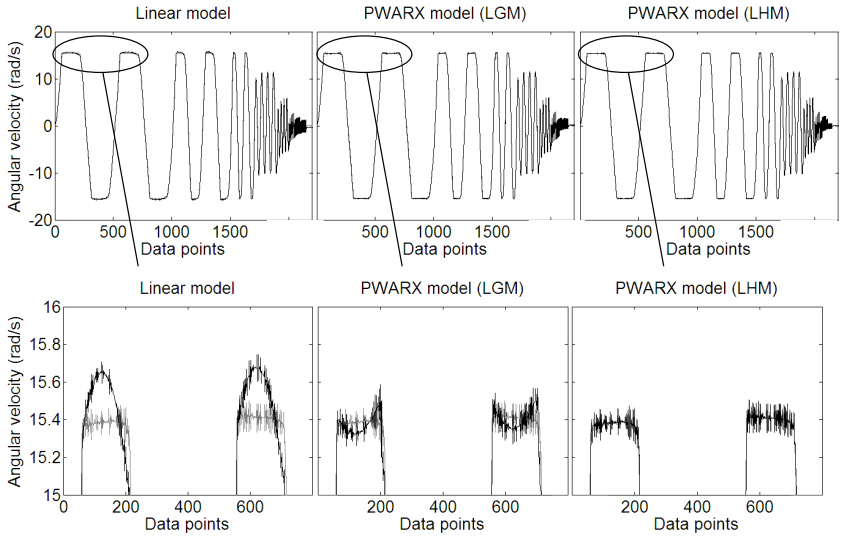

Figure 3.12: Data fitting for the training data. Gray: measured, black: estimated.

identified for comparison:

$$\hat{v}(t+1) = \theta_1 v(t) + \theta_2 u(t) + \theta_3 \tag{3.41}$$

The identification result is shown in Fig. 3.12. It is clear that a single linear model is not able to capture the piecewise affine characteristic of the motor velocity, whereas the PWARX models, identified using the proposed algorithms, could provide a much better data fitting.

To evaluate the fit between the measured and the estimated velocity, the following "Fit" value is calculated:

$$\text{Fit} = \left(1 - \sqrt{\frac{\sum_t (v(t+1) - \hat{v}(t+1))^2}{\sum_t (v(t+1) - \bar{v})^2}} \right) \times 100\% \tag{3.42}$$

where $\bar{v}$ is the arithmetic mean of $v(t+1)$. The result is shown in TABLE 3.8. It is noticed that the model identified using the LHM algorithm is slightly more superior than that obtained using the LGM algorithm, because the LHM is less sensitive to measurement noise.

Table 3.8: Fit values of the identified models for DC motor

Model	Linear	PWARX (LGM)	PWARX (LHM)
Training data	97.3%	97.8%	98.0%
Test data 1	96.8%	98.2%	98.7%
Test data 2	97.8%	98.3%	98.6%

The model (3.40) identified using the LHM algorithm is:

$$\hat{v}(t+1) = \begin{cases} 0.948v(t) + 0.283u(t) + 0.022 & \text{if } [v(t), u(t)] \in X_1 \\ 0.999v(t) + 0.001u(t) + 0.000 & \text{if } [v(t), u(t)] \in X_2 \end{cases} \tag{3.43}$$

where the numerical values of the second subsystem clearly show that the velocity stays almost constant and is unaffected by the control input, once the velocity saturation is reached.

Validation for the Test Sets

After the parameters of the models have been identified using the training data, the ability of the models to generalize on data which was not seen during the training phase is tested. For this purpose, two types of input signals are defined, the first being a stair-type signal, with magnitudes from 0.5V to 5V in the increments of 0.5V, and the second being a sinusoidal signal with 1V amplitude and 1Hz frequency, superimposed with a stair-type signal with magnitudes 0V, 2V and 4V.

The validation results are shown in Figs. 3.13 and 3.14 respectively. Again, it is seen that the single linear model cannot provide a good prediction of the velocity after saturation level is reached, whereas the PWARX models have no difficulty in capturing the PWA aspects of the motor, with the one identified using the LHM algorithm being more accurate, as shown in TABLE 3.8.

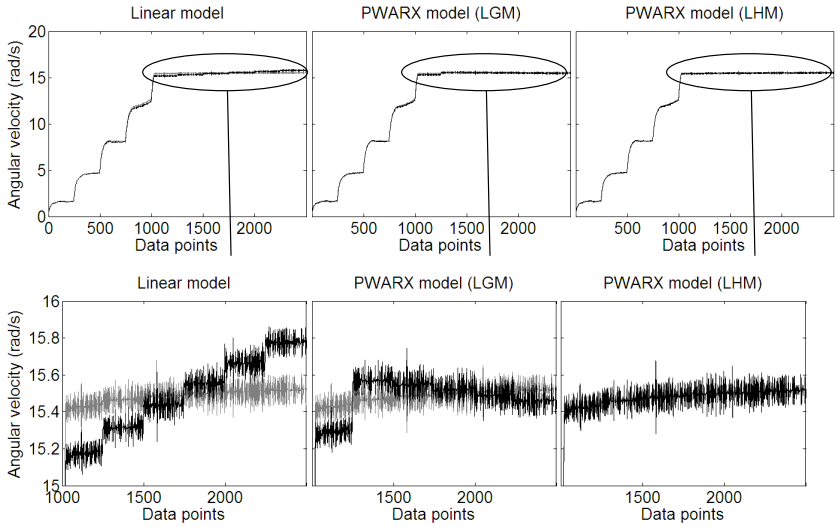

Figure 3.13: Data fitting for the test data 1. Gray: measured, black: estimated

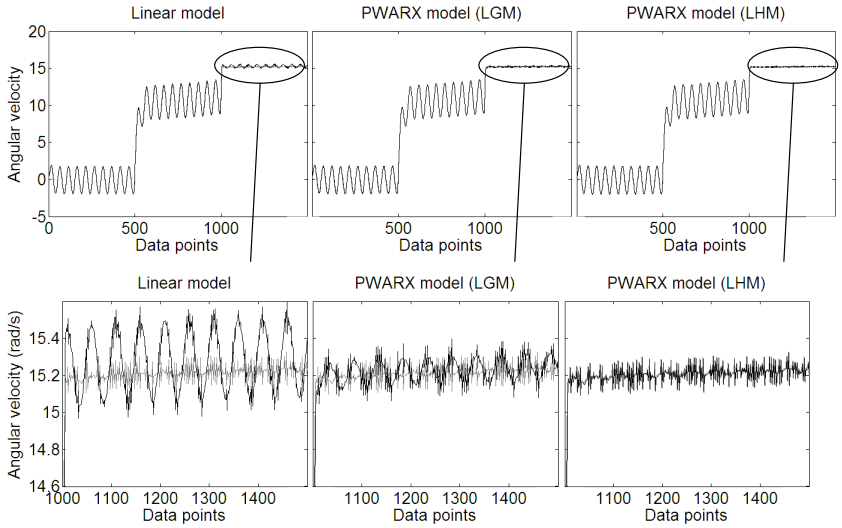

Figure 3.14: Data fitting for the test data 2. Gray: measured, black: estimated

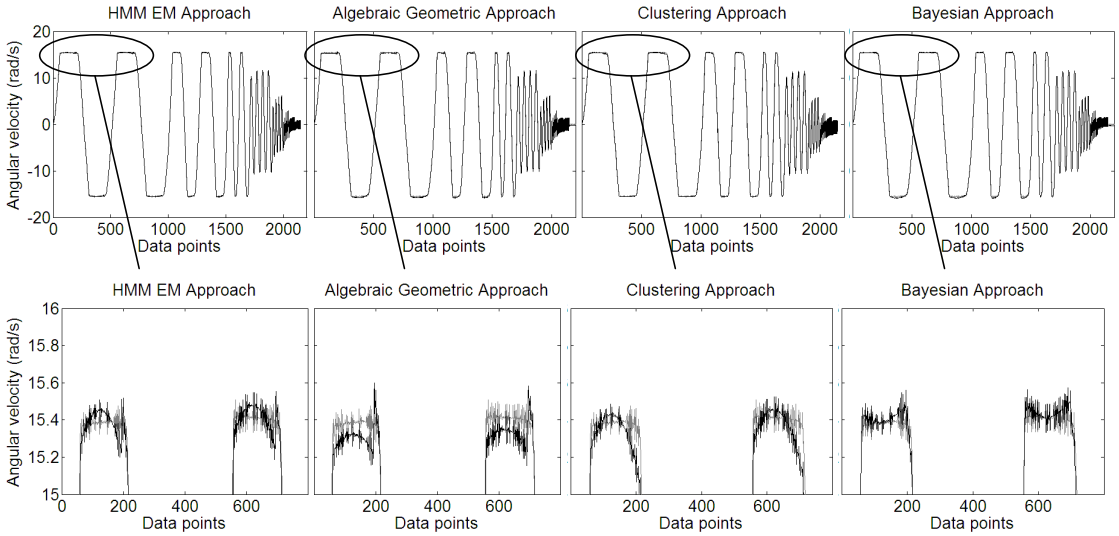

Figure 3.15: Data fitting for the training data using other algorithms. Gray: measured, black: estimated.

Comparison with other Algorithms

The proposed method is compared with four other algorithms found in the literature, namely the Hidden Markov Model plus Baum-Welch algorithm (HMM EM) of [24], the Algebraic-Geometric method of [77], the Clustering technique of [20], and the Bayesian approach of [29]. The tuning parameters of each of these algorithms (for e.g. the number of data points per cluster c in the clustering technique) were carefully chosen so as to obtain the best result. The results for the training data and the test data 1 and 2 are shown in Figs. 3.15 to 3.17 respectively.

Comparing Figs. 3.15 to 3.17 with Figs. 3.12 to 3.14, it can be observed that the proposed algorithms, especially the least harmonic mean (LHM) algorithm, outperform the other compared methods. This conclusion is also supported by the Fit values as shown in TABLE 3.9.

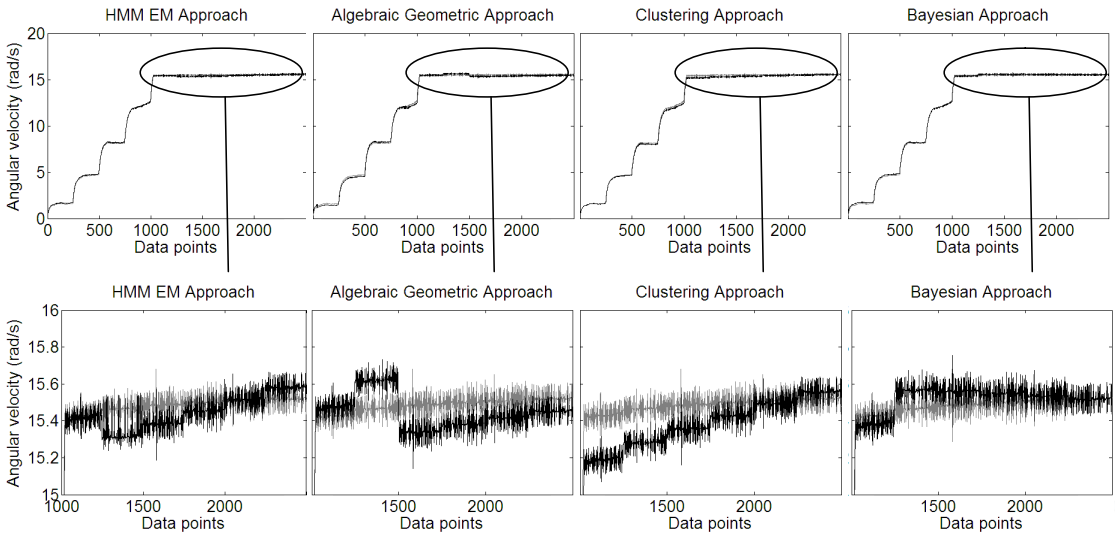

Figure 3.16: Data fitting for the test data 1 using other algorithms. Gray: measured, black: estimated.

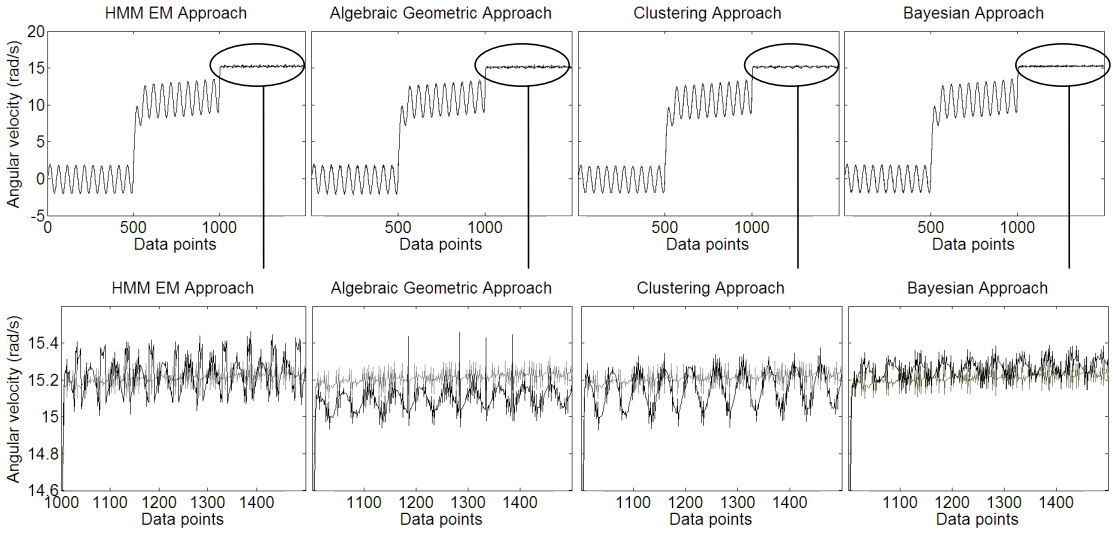

Figure 3.17: Data fitting for the test data 2 using other algorithms. Gray: measured, black: estimated.

Table 3.9: Fit values of the identified models using other algorithms

Algorithm	HMM EM	Algebraic	Clustering	Bayesian
Training data	98.1%	97.6%	96.7%	97.6%
Test data 1	98.1%	97.2%	97.2%	98.1%
Test data 2	98.4%	97.8%	97.7%	98.1%

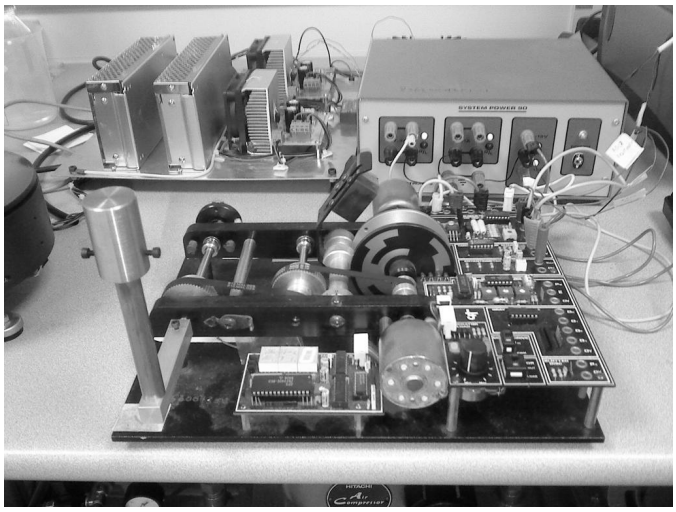

Figure 3.18: Hardware setup of the single-link robotic arm

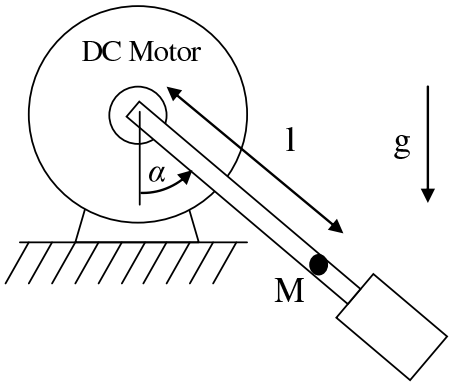

Figure 3.19: Schematics diagram of the single-link robotic arm

3.7.2 Single Link Robotic Arm

A DC motor apparatus by L. J. Electronics, which is used as a teaching set for teaching control courses, has been slightly modified to be a single link robotic arm. Specifically, the original load which is centric, has been replaced by a brass rod with a heavy brass pendulum at the end. The setup is shown in Fig. 3.18, and its schematic diagram is shown in Fig. 3.19.

The simplified physical model of the system is

$$J\ddot{\alpha} = -\beta\dot{\alpha} - Mgl\sin(\alpha) + Ku \tag{3.44}$$

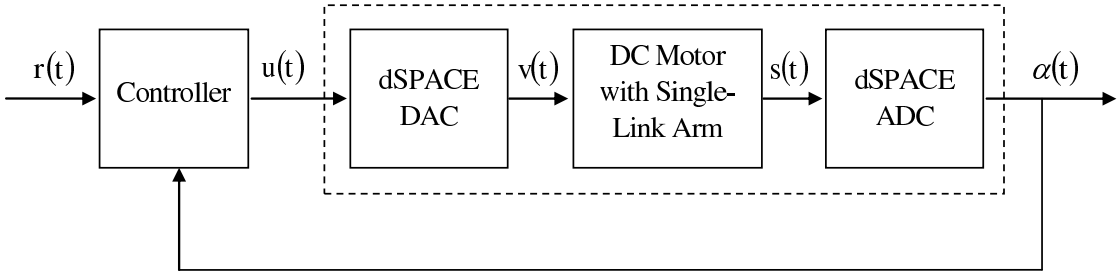

Figure 3.20: The hardware-in-the-loop simulation for the single-link robotic arm

where J is the overall moment of inertia of the system including the motor shaft, the rod and the pendulum, M is the mass of the rod and the pendulum, l is the distance from the pivot of rotation to the center of gravity of the rod and pedulum, g is the gravitational constant, β is the damping coefficient, α is the angle of rotation, and u is the input voltage. Determining l and β is not very straightforward, and thus it could be advantageous to use a data-based approach to identify the system.

PWARX Model of the Robotic Arm

Because the system has an integrator, and since we do not want the robotic arm to rotate endlessly (or the angle to integrate up to a huge number), we first stabilize the system using a PD-type controller. Note that this PD-type controller does not need to be tuned painstakingly to achieve excellent tracking performance. It merely serves to stabilize the system and to allow the acquiring of input-output data.

The PD Controller is programmed in MATLAB/Simulink, and a dSPACE DS1104 rapid control prototyping system is used to generate the physical control signal $v(t)$ to the DC Motor, based on the calculated numerical value of the control signal $u(t)$. Also, the sensor reading of the angle $s(t)$ is sent into the computer via the dSPACE system, and be interpreted as numerical values $\alpha(t)$. The working diagram is shown in Fig. 3.20.

For this experiment, we chose the sampling rate to be 10Hz. The input-output

PWARX model of the system is then identified based on the signals $u(t)$ and $\alpha(t)$, and the PWARX model is of the form

$$\alpha(t+1) \approx \begin{cases} \varphi^T(t)\theta_1^* & \text{if } \tilde{x}(t) \in \tilde{X}_1 \\ \vdots & \vdots \\ \varphi^T(t)\theta_N^* & \text{if } \tilde{x}(t) \in \tilde{X}_N \end{cases} \tag{3.45}$$

where

$$x(t) = [\alpha(t), \alpha(t-1), u(t), u(t-1)]^T \tag{3.46}$$

$$\varphi(t) = \left[x(t)^T, 1\right]^T \tag{3.47}$$

$$\tilde{x}(t) = [\alpha(t), \alpha(t-1), u(t-1)]^T \tag{3.48}$$

The parameters of the two subsystems are identified using the proposed least harmonic mean squares (LHM) algorithm, whereas the regions $\tilde{X}_i$ are estimated using multilayer perceptrons.

Training Set

The training set, which consists of 4000 data points, was obtained by setting the reference signal to be

$$r(t) = 0.6\sin(2\pi 0.01t) + 0.5\sin(2\pi 0.1t) + 0.3\sin(2\pi 0.4t + \pi/3) + \pi + \epsilon(t) \tag{3.49}$$

where $\epsilon(t)$ is a random number in the range of [-0.4, 0.4]. This randomness is introduced so that the data set is richer in frequency content.

The identification errors for the training set are shown in Fig. 3.21. As can be seen, the error decreases as we have more subsystems in the PWARX model.

To have a more quantitative evaluation of the accuracy of the identified model, we compute the "Fit"-value which is defined as

$$\text{Fit} = \left(1 - \sqrt{\frac{\sum_t (\alpha(t+1) - \hat{\alpha}(t+1))^2}{\sum_t (\alpha(t+1) - \bar{\alpha})^2}}\right) \times 100\% \tag{3.50}$$

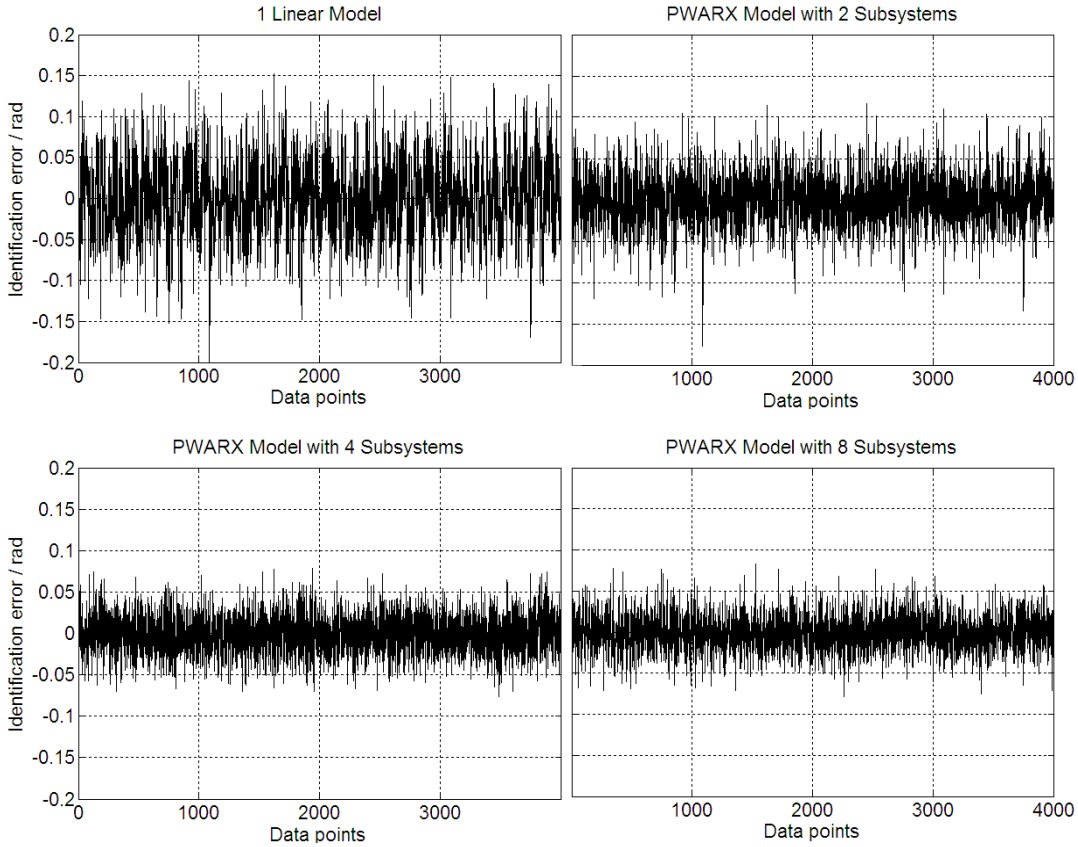

Figure 3.21: Identification error for the training set

where $\alpha(t+1)$, $\hat{\alpha}(t+1)$ and $\bar{\alpha}$ are the true output, estimated output, and the average of $\alpha(t+1)$ respectively. The result is shown in TABLE 3.10. The values show that the data-fitting improves as the number of subsystems increases.

Table 3.10: Fit values for single link robotic arm

No. subsystems	1	2	4	8
Training data	96.26%	97.49%	98.14%	98.33%
Test data	96.37%	97.59%	98.14%	98.49%

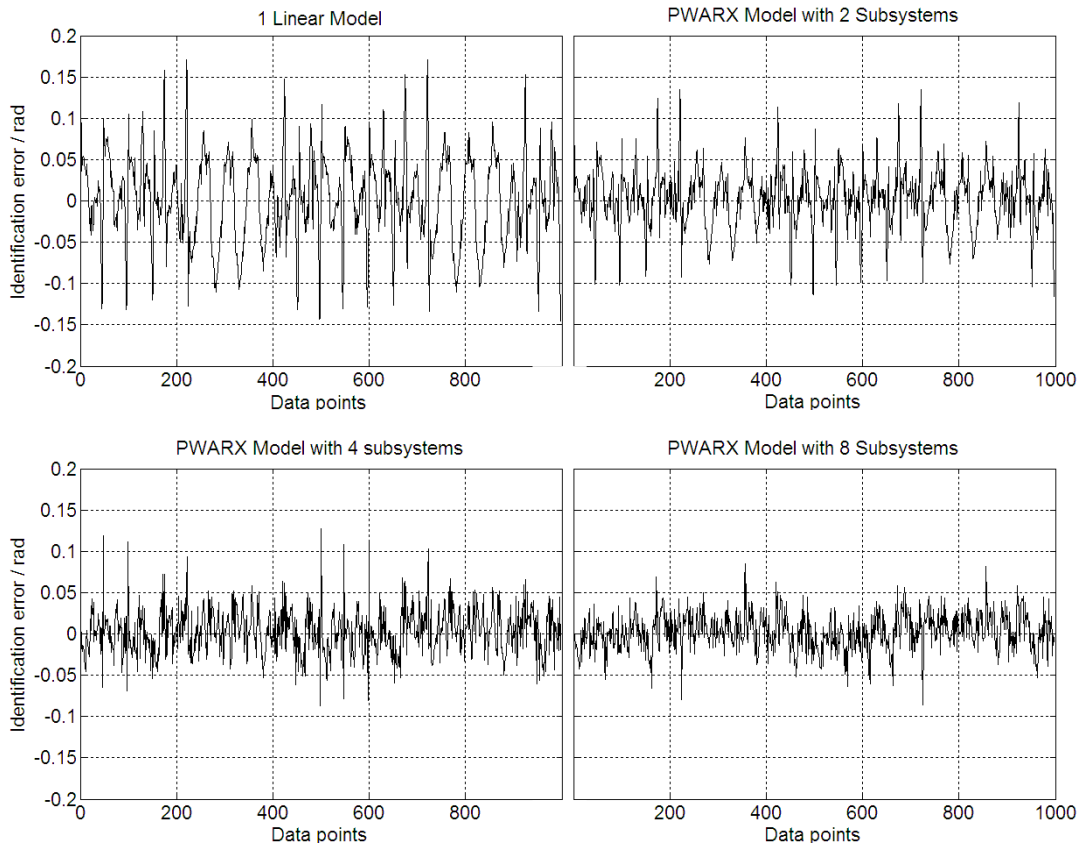

Figure 3.22: Identification error for the test set

Test Set

The test set, which contains 1000 data points, was obtained by setting the reference signal to be

$$r(t) = 0.7\sin(2\pi 0.02t) + 0.8\sin(2\pi 0.2t) + \pi \tag{3.51}$$

The identification errors for the test set are shown in Fig. 3.22. As the number of subsystems in the PWARX model increases, the identification error decreases. This shows that the PWARX Models can generalize well to the test data.

3.8 Conclusions

In this chapter, a procedure for the identification of piecewise affine ARX systems was proposed. The proposed algorithm consists of two steps: identification of the parameters of the subsystems and estimation of the partition of the regressor space. In the first step of the algorithm, the parameters of the individual subsystems were identified through a least-squares based identification method using multiple models, without needing to know the boundaries of the partitions. This step has been handled in chapter 2. The partition of the regressor space was then determined in the second step using standard procedures such as neural network classifier or support vector machine classifier. We also used the same procedure to identify nonlinear systems by approximating them via piecewise affine systems. Extensive simulation studies show that our algorithm can indeed provide accurate estimates of the plant parameters even in noisy cases, and even when the model orders are overestimated. For experimental verification, the method was used to obtain a PWARX model of an electric motor which shows a velocity saturation characteristic. The result showed that the proposed algorithm indeed provided an accurate model of the electric motor and could capture its piecewise affine nature. The method was also employed to obtain a model of a single-link robotic arm, where it was shown that the PWARX model provides a better description of the robotic arm than by using a single linear model.

Chapter 4

Control of Piecewise Affine Systems and Nonlinear Systems using Multiple Models

4.1 Introduction

As mentioned in chapter 1, one approach of controller design for nonlinear systems, which has a large following in the industry as well as in flight control, is to first separate the operating region of the nonlinear process into several subregions, design a linear controller for each of these partitions, and then activate the controller corresponding to the active operating condition at a certain time. The main advantages of this approach are its simplicity and intuitive appeal, and that powerful linear design tools can be employed on difficult nonlinear problems.

The partitioning of the full operating region into several smaller subregions has resulted in modeling formalisms of piecewise-linear or piecewise-affine (PWA) models, and several work have appeared in the literature to deal with the control design for these PWA systems. In [25, 63–66], the controllers were designed for quadratic stabilization of the PWA systems, with the help of piecewise quadratic Lyapunov functions as well

as convex optimization. The PWA systems considered in these works are in state space forms, but the authors showed that it was possible to use state observers to estimate the states if only the output signal is available. In [12], a dynamic controller is designed based on the common Lyapunov function. Although it is known that the common Lyapunov function provides a more conservative controller than the piecewise Lyapunov function, the authors were able to show that regional pole placement of the closed loop system can be achieved. [80] designed dynamic controllers based on pole placement for each of the linear subsystems, and then analysed the conditions for stability of the closed loop system under switching. Switching H-infinity controllers for a class of linear parameter varying systems are designed in [83]. Last but not least, [27] designed optimal control for sampled-data piecewise affine systems, i.e. systems whose switching action of the discrete state is determined at each sampling time according to a condition of the continuous state.

The by far most popular control methodology for piecewise affine systems, switching systems and hybrid systems is the model predictive control [1–5, 7–9, 14, 17, 22, 23, 45, 51, 52, 58, 59, 62, 76]. As in the usual model predictive controller for a single model system, the control signal is computed by minimizing a cost function which penalizes the future output error and the control signal. However, an additional point which needs to be taken into consideration is the switching of the model's subsystems and the corresponding local controllers. By viewing the activation and deactivation of the local controllers as a discrete state of one or zero, some researchers proposed to include this discrete state in the cost function, and solve a mixed-integer linear program or mixed-integer quadratic program [7,8]. Due to the complexity of the mixed-integer programming, these controllers might have difficulties to be implemented in real time. As such, [4, 5, 45] proposed to

recast the optimization problem as multiparametric mixed-integer programming, which is first solved offline, and the optimal control signal is obtained as an explicit function of the states. Then, in real time implementation, the optimal control signal is obtained merely via a simple function evaluation. Unfortunately, this method was demonstrated in depth only for the Mixed Logical Dynamical (MLD) model, which although is proven to be equivalent to piecewsie affine systems in [26], is less intuitive and not easy to comprehend.

Another issue related to the control of nonlinear systems by patching up multiple local controllers is how to determine/design the switching among the linear controllers. In [47,48], the "supervisor" determines the best local controller to be used at a particular instant by evaluating certain norm-squared output estimation errors of the local models. [36], on the other hand, evaluates the best controller to be activated by comparing the "virtual" closed-loop performance. Yet, a complete switching between the controllers is not the only way to utilize the multiple model controllers. It is also possible to weigh the output of each controllers based on some fuzzy or bayesian rules and sum them up as the final control signal. This approach was used in [2, 3, 51].

In this chapter, we will derive a model-based controller to control piecewise affine systems and nonlinear systems for reference tracking, the latter being approximated using piecewise affine ARX models. The local controllers are weighted one-step-ahead controllers, and they switch from one to another by observing the active partition of the regressor space. This is possible because we have already trained data classifiers to estimate the regressor space partition. However, there also exists a difficult chicken-and-egg problem, where the determination of the active subsystem needs the information of the yet-to-be-calculated control signal. We will therefore propose a few solutions to

overcome this difficulty. Both simulation and experimental studies verified the efficacy of our control algorithm.

4.2 Weighted One-Step-Ahead Controller

The model of the PWARX system

$$\hat{y}(t+1) = \begin{cases} \varphi^T(t)\theta_1 & \text{if } x(t) \in X_1 \text{ or } \tilde{x} \in \tilde{X}_1 \\ \vdots & \vdots \\ \varphi^T(t)\theta_m & \text{if } x(t) \in X_m \text{ or } \tilde{x} \in \tilde{X}_m \ (m \in (1,\ldots,N)) \\ \vdots & \vdots \\ \varphi^T(t)\theta_N & \text{if } x(t) \in X_N \text{ or } \tilde{x} \in \tilde{X}_N \end{cases} \tag{4.1}$$

where

$$x(t) = [y(t),\ldots,y(t-n_a),u(t),\ldots,u(t-n_b)]^T \tag{4.2}$$

$$\varphi(t) = \left[x^T(t),1\right]^T \tag{4.3}$$

$$\tilde{x}(t) = [y(t),\ldots,y(t-n_a),u(t-1),\ldots,u(t-n_b)]^T \tag{4.4}$$

and

$$\cup_{i=1}^{N} X_i = X \text{ and } X_i \cap X_j = \emptyset,\ \forall i \neq j \tag{4.5}$$

where X denotes the whole regressor space, can be explicitly expressed in the following form:

$$\hat{y}(t+1|t) = \begin{cases} a_{1n_a}y(t) + \cdots + a_{10}y(t-n_a) & \\ +b_{1n_b}u(t) + \cdots + b_{10}u(t-n_b) + c_{10} & \text{if } x(t) \in X_1 \text{ or } \tilde{x} \in \tilde{X}_1 \\ \vdots & \\ a_{Nn_a}y(t) + \cdots + a_{N0}y(t-n_a) & \\ +b_{Nn_b}u(t) + \cdots + b_{N0}u(t-n_b) + c_{N0} & \text{if } x(t) \in X_N \text{ or } \tilde{x} \in \tilde{X}_N \end{cases} \tag{4.6}$$

This can also be unified in the form of

$$\begin{aligned}\hat{y}(t+1|t) = a_{\sigma(t),n_a}y(t) + \cdots + a_{\sigma(t),0}y(t-n_a) \\ + b_{\sigma(t),n_b}u(t) + \cdots + b_{\sigma(t),0}u(t-n_b) + c_{\sigma(t),0}\end{aligned} \tag{4.7}$$

where $a_{\sigma(t),i}$, $b_{\sigma(t),i}$ and $c_{\sigma(t),0}$ are time-varying parameters which depend on the active region $X_1, \ldots, X_N$ or $\tilde{X}_1, \ldots, \tilde{X}_N$. Here, $\sigma(t)$ denotes the active subsystem, i.e.

$$\sigma(t) = m \text{ if } x(t) \in X_m \text{ or } \tilde{x}(t) \in \tilde{X}_m \tag{4.8}$$

The one-step-ahead control law which would ensure that $\hat{y}(t+1)$ tracks the reference $r(t+1)$ is

$$u(t) = \frac{\begin{bmatrix} r(t+1) \\ -a_{\sigma(t),n_a}y(t) - \cdots - a_{\sigma(t),0}y(t-n_a) \\ -b_{\sigma(t),n_b-1}u(t-1) - \cdots - b_{\sigma(t),0}u(t-n_b) \\ -c_{\sigma(t),0} \end{bmatrix}}{b_{\sigma(t),n_b}} \tag{4.9}$$

This is obtained by minimizing $(r(t+1) - \hat{y}(t+1))^2$.

To improve the stability of the one-step-ahead controller, we can also use the weighted one-step-ahead control law as follows

$$u(t) = \frac{b_{\sigma(t),n_b}\begin{bmatrix} r(t+1) \\ -a_{\sigma(t),n_a}y(t) - \cdots - a_{\sigma(t),0}y(t-n_a) \\ -b_{\sigma(t),n_b-1}u(t-1) - \cdots - b_{\sigma(t),0}u(t-n_b) \\ -c_{\sigma(t),0} \end{bmatrix}}{b^2_{\sigma(t),n_b} + \lambda} \tag{4.10}$$

where λ is a nonnegative number. The weighted one-step-ahead control law can be obtained by minimizing the following cost function

$$J = (\hat{y}(t+1) - r(t+1))^2 + \lambda u^2(t) \tag{4.11}$$

with respect to $u(t)$. λ can thus be seen as a tuning parameter to achieve a balance between tracking accuracy and control magnitude.

4.3 A Chicken-and-Egg Situation and its Solutions

Note that the computation of the control signal $u(t)$ according to (4.9) or (4.10) depends on the switching signal $\sigma(t)$. We however run into a rather tricky "chicken and egg" situation here, if we use the standard regressor $x(t)$: At time t, we need the regressor $x(t) = [y(t), \ldots, y(t-n_a), u(t), \ldots, u(t-n_b)]^T$ to decide the current switching signal $\sigma(t) = h(x(t))$ using the data classifier, but $u(t)$ within $x(t)$ is yet to be designed at this stage!

To overcome this akward situation, we have developed five possible methods, which are listed next from the worst performance to the best.

4.3.1 Method I: Using the Previous Switching Signal

The simplest solution is to assume that the switching is not so frequent, such that $\sigma(t) = \sigma(t-1)$, i.e. we will use $x(t-1) = [y(t-1), \ldots, y(t-n_a-1), u(t-1), \ldots, u(t-n_b-1)]^T$ to determine the active subsystem at time t.

The advantage of this scheme is its simplicity, but its obvious drawback is that the assumption can be violated from time to time when switching actually occurs. This leads to extremely poor performance if switching occurs frequently, or when the nonlinearity of the system is severe.

4.3.2 Method II: Compute $u(t)$ for all possible switching signals and compare the cost functions

The second method is to compute the control signals for all possible switching signals $\sigma(t)$. The computed control signals are then substituted into the cost function $J = (\hat{y}(t+1) - r(t+1))^2 + \lambda u^2(t)$, and the corresponding signal which gives the lowest cost value will be used.

This method has the advantage that it conforms with the spirit of cost function optimization (minimization). However, it still does not guarantee a good control performance, because the subsystem with the minimum cost value is not necessarily the true active subsystem! By using the corresponding incorrect control signal, the system output will not track the reference as desired.

4.3.3 Method III: Compute $u(t)$ for all possible switching signals and check the active subsystem

The third method is as follows: First, we shall assume $\sigma_{\text{assumed}}(t) = 1$, and compute $u_1(t)$ according to (4.9) or (4.10). We then feed our data classifier with $x(t) = \Big[y(t), \ldots, y(t-n_a), \underline{u_1(t)}, u(t-1), \ldots, u(t-n_b)\Big]^T$, and check whether $\sigma(t)$ is indeed 1. If this is true, it would mean that we have made a correct guess of $\sigma(t)$, and we can send this control signal $u(t) = u_1(t)$ to the system.

If however $\sigma(t) \neq \sigma_{\text{assumed}}(t) = 1$, we should proceed to the next step instead, as follows: We now assume $\sigma_{\text{assumed}}(t) = 2$, and calculate $u_2(t)$ according to (4.9) or (4.10). Letting the input of the data classifier to be $x(t) = \Big[y(t), \ldots, y(t-n_a), \underline{u_2(t)}, u(t-1), \ldots, u(t-n_b)\Big]^T$, we check whether $\sigma(t)$ indeed equals 2. If this is correct, it means we have correctly assumed the active subsystem, and we can proceed to feed the process with $u(t) = u_2(t)$.

If this is also not true, we proceed similarly for subsystems 3 to N, until one of the $\sigma_{\text{assumed}}(t)$ equals the correct $\sigma(t)$, and use the corresponding control signal.

In case none of the above is correct, we might need to resort to one of the earlier two methods, but the tracking performance will definitely be poor at these instances.

4.3.4 Method IV: Engage the data classifier while computing $u(t)$

This is a refinement from Method III. There, we had no control on whether $u_1(t)$ will make subsystem 1 active, $u_2(t)$ will make subsystem 2 active and so on. As mentioned, if none of these are true, the control performance will degrade.

To improve the situation, we will not calculate the control signals using (4.9) or (4.10) alone. Instead, we shall compute the control signal by solving the following constrained optimization problem

$$\begin{aligned}
&\min_{u(t)} J = (\hat{y}(t+1) - r(t+1))^2 + \lambda u^2(t) \\
&\qquad = \left(\begin{array}{l} a_{\sigma(t),n_a} y(t) + \cdots + a_{\sigma(t),0} y(t-n_a) \\ +b_{\sigma(t),n_b} u(t) + \cdots + b_{\sigma(t),0} u(t-n_b) + c_{\sigma(t),0} - r(t+1) \end{array} \right)^2 + \lambda u^2(t) \\
&\text{s.t. } \sigma(t) = h([y(t), \ldots, y(t-n_a), u(t), \ldots, u(t-n_b)]) = \sigma_{\text{predefined}}
\end{aligned} \tag{4.12}$$

whereby the function $h([y(t), \ldots, y(t-n_a), u(t), \ldots, u(t-n_b)])$ is obtained from the data classifier.

The procedure is applied to all possible switching signals ($\sigma_{\text{predefined}}$). Some of the switching signals will not provide feasible solutions (i.e. there is no $u(t)$ which, together with the regression vector, would make the predefined subsystem active) and they can be discarded. For those which are feasible, the control signals are then substituted into the cost function and the one corresponding to the least cost will be used.

The advantage of this scheme is good control performance, because the correct active subsystem will definitely be used when calculating the control signal. However, computational burden in solving the (nonlinear) constrained optimization problem can prohibit the use of this method in real-time problem.

4.3.5 Method V: Ad-Hoc scheme using Classifier II

It was due to the difficult chicken-and-egg situation and the lack of good solution which inspired the development of classifier II, where $u(t)$ is ignored in the data classification process. By using $\tilde{x}(t) = [y(t), \ldots, y(t-n_a), u(t-1), \ldots, u(t-n_b)]^T$, the problem is circumvent. Even though one might argue that this classifier might not provide good identification results, most of our simulations show that this is still a reliable scheme. Most importantly, the control performance is superior to when the chicken-and-egg problem needs to be tackled.

Remark 4.1 *It is clear that stability is an important issue when designing control systems. However, a stability analysis for the control of nonlinear systems using multiple linear controllers is a very challenging problem. For instance, it is well-known that even though each of the local linear controllers can stabilize its corresponding approximately-linear region of the system, stability can still be lost if there is frequent switching among the subsystems. Also, although there are some stability results on model predictive controller for hybrid systems, they all assume that the model is accurate. In our case, since the PWARX model is an approximate model of the original nonlinear system, the stability results are not directly applicable. These issues will be investigated as part of our future work.*

4.4 Simulation Studies

4.4.1 Nonlinear System 1

Consider the following nonlinear system,

$$y(t+1) = \frac{y(t)}{1+y^2(t)} + u^3(t) \tag{4.13}$$

which was identified in section 3.6.3. There, we showed that by using the PWARX model of the form

$$y(t+1) \approx \begin{cases} \varphi^T(t)\theta_1^* & \text{if } x(t) \in X_1 \\ \vdots & \vdots \\ \varphi^T(t)\theta_N^* & \text{if } x(t) \in X_N \end{cases} \tag{4.14}$$

$$\text{where} \qquad x(t) = [y(t), u(t)]^T$$

$$\varphi(t) = \left[x(t)^T, 1\right]^T$$

and by using 4,8 or 16 models, the input-output data can be fit to a reasonable accuracy.

We now proceed to control the system using the weighted one-step-ahead control law (4.10). The nonlinear system is required to track the reference signal

$$r(t) = 4\sin(2\pi t/10) + 4\sin(2\pi t/25) \tag{4.15}$$

The simulation result for the control of the nonlinear system using Method III is shown in Fig. 4.1. As can be observed, the nonlinear system can track the reference signal rather well using the controller which is designed based on the PWARX models with 2, 4 and 16 subsystems, as compared to the case when only one linear model is used.

However, we also notice that sometimes, the tracking is not too accurate. This is because the control signals were calculated according to the "assumed" active subsystem but none of the computed control signal makes the corresponding subsystem active. Therefore, we proceed to try Method IV, i.e. by engaging the data classifier while performing the cost function optimization. A comparison of Method III and Method IV for the case of using the PWARX model with 4 subsystems is shown in Fig. 4.2.

As can be seen, the tracking performance is significantly improved by using Method IV, because the control signal is calculated with the constraint that it will necessarily make the predefined subsystem to be active.

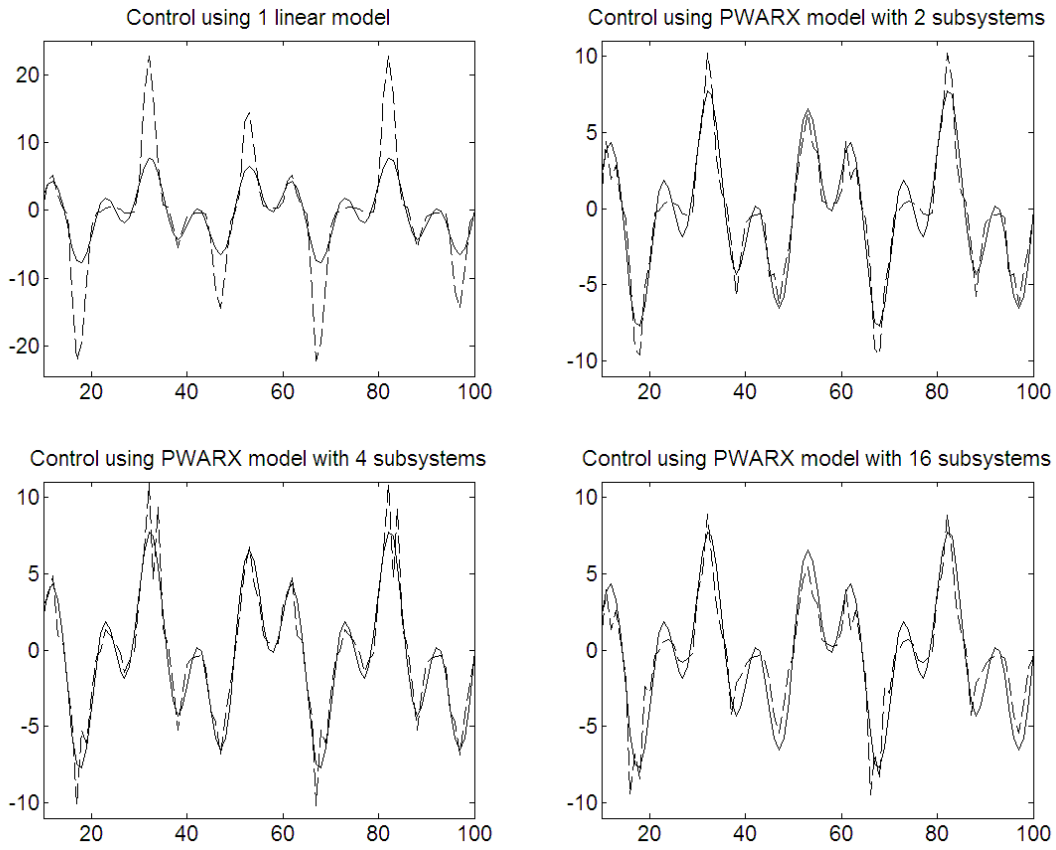

Figure 4.1: Control of nonlinear system 1 - Method III.

Solid: Reference, dashed: Output of nonlinear system.

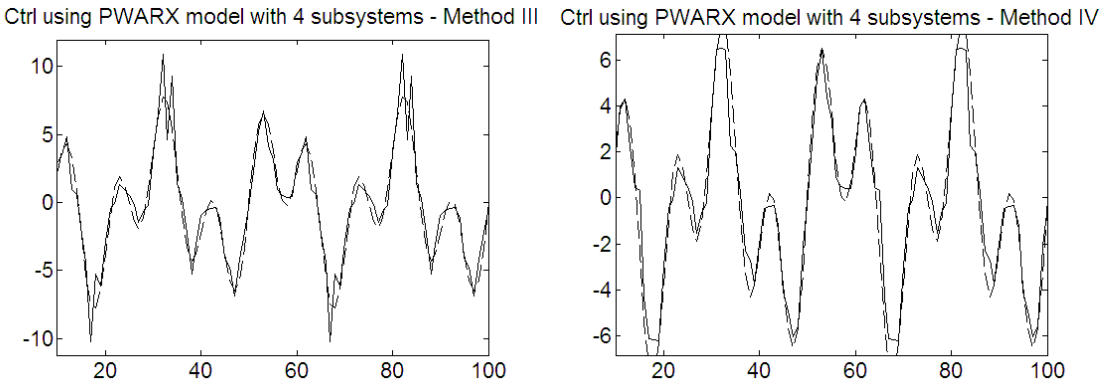

Figure 4.2: Control of nonlinear system 1 - Method III vs. Method IV.

Solid: Reference, dashed: Output of nonlinear system.

4.4.2 Nonlinear System 2

Consider next the nonlinear system which was previously identified in section 3.6.4

$$\begin{aligned} y(t+1) = & \frac{1.5y(t)y(t-1)}{1+y^2(t)+y^2(t-1)} \\ & + \sin(y(t)+y(t-1)) + u(t) + 0.8u(t-1) \end{aligned} \tag{4.16}$$

The nonlinear system was approximated using the PWARX model

$$y(t+1) \approx \begin{cases} \varphi^T(t)\theta_1^* & \text{if } x(t) \in X_1 \text{ or } \tilde{x}(t) \in \tilde{X}_1 \\ \vdots & \vdots \\ \varphi^T(t)\theta_N^* & \text{if } x(t) \in X_N \text{ or } \tilde{x}(t) \in \tilde{X}_N \end{cases} \tag{4.17}$$

where

$$x(t) = [y(t), y(t-1), u(t), u(t-1)]^T \tag{4.18}$$

$$\varphi(t) = \left[x(t)^T, 1\right]^T \tag{4.19}$$

$$\tilde{x}(t) = [y(t), y(t-1), u(t-1)]^T \tag{4.20}$$

It was shown that both Classifier I and Classifier II provided similar identification results, and that the data could be fit well by using a PWARX model with 2, 4 or 8 subsystems.

With an accurate PWARX model of the nonlinear system, we proceed to control the system using the one-step-ahead control law (4.9). The nonlinear system is required to track the reference signal

$$r(t) = \sin(2\pi t/10) + \sin(2\pi t/50) \tag{4.21}$$

The simulation result for the control of the nonlinear system using Method I is shown in Fig. 4.3. The performance of the controller is very poor. This can be attributed to the fact that the assumption $\sigma(t) = \sigma(t-1)$ does not hold well. The wrong subsystem has been assumed at time t, and thus the control signal is computed incorrectly.

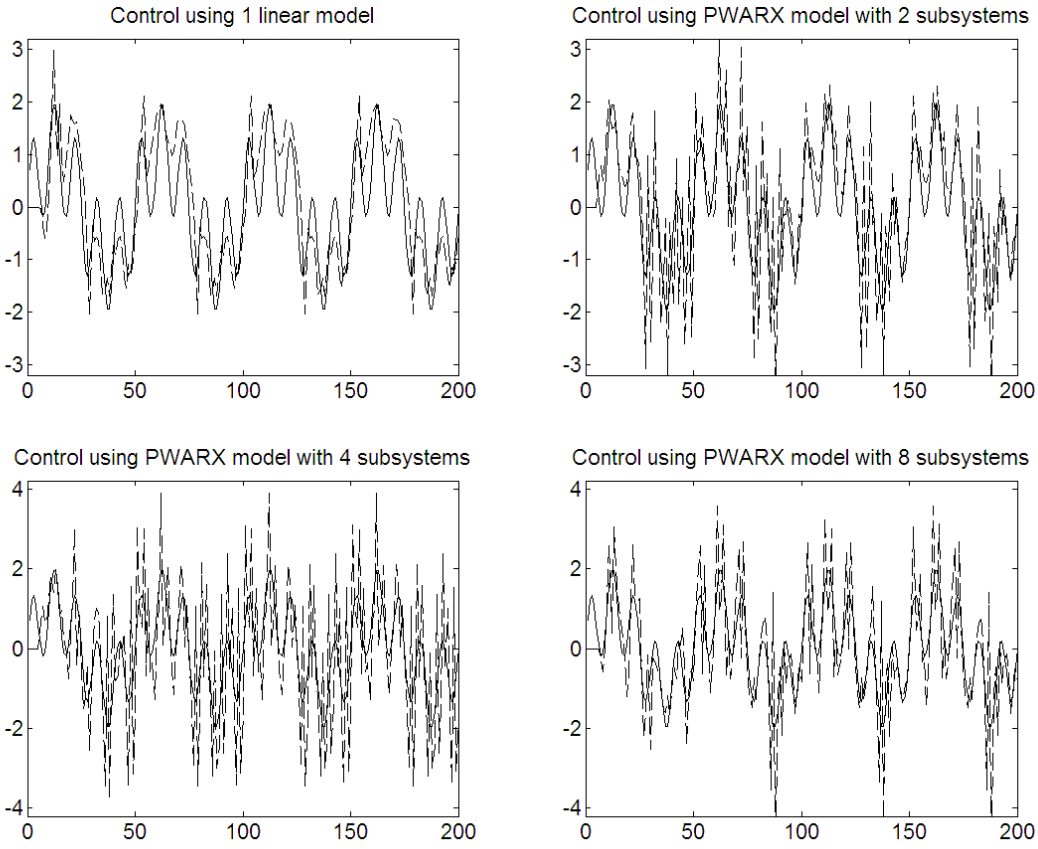

Figure 4.3: Control of the nonlinear system 2 - Method I.

Solid: Reference, dashed: Output of nonlinear system.

We then proceed to use Method IV to control the nonlinear system. The simulation result using Method IV is shown in Fig. 4.4. As can be observed, the control performance improved significantly, as compared to Method I.

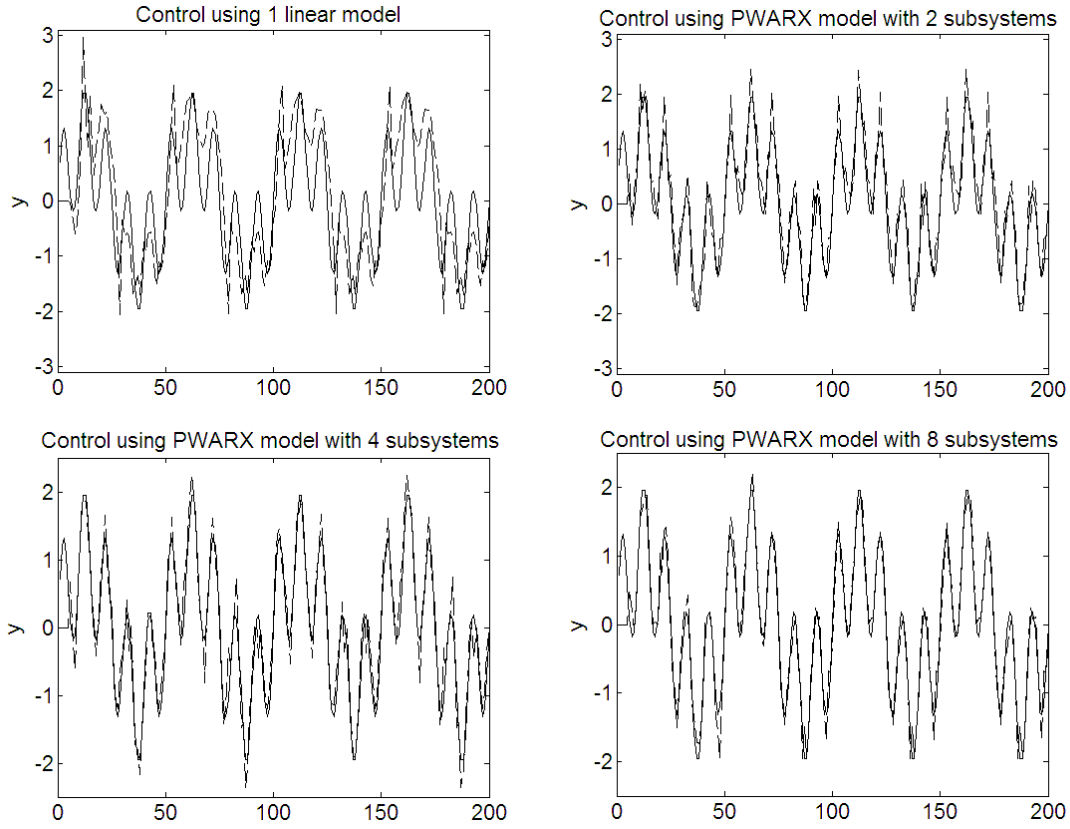

Figure 4.4: Control of the nonlinear system 2 - Method IV.
Solid: Reference, dashed: Output of nonlinear system.

However, as mentioned, this method has a large computational burden and is not implementable in real-time. For instance, for the case of using the PWARX model with 8 subsystems, the simulation time was more than two minutes for the 200 data points on a computer with Intel(R) Core(TM)2 Duo CPU (2.53GHz) and 2.95 GB RAM at 1.59 GHz.

The simulation result for the control performance using Method V is shown in Fig. 4.5. As can be observed, the nonlinear system can track the reference signal rather

well using the controller which is designed based on the PWARX model with 4 or 8 subsystems. Most importantly, the simulation time was very short, and was only two seconds for the case with 8 subsystems.

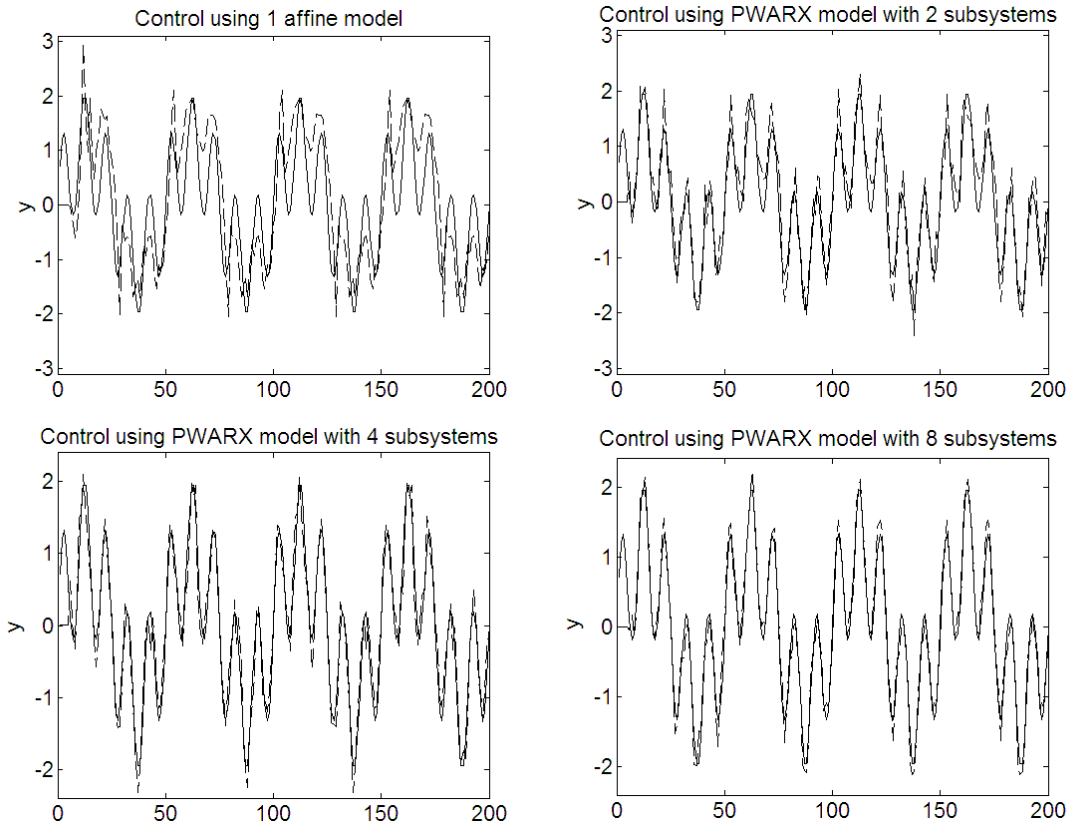

Figure 4.5: Control of the nonlinear system 2 - Method V.
Solid: Reference, dashed: Output of nonlinear system.

4.4.3 Nonlinear System 3

Consider again the nonlinear benchmark system given in the state space form:

$$\begin{aligned}
\xi_1(t+1) &= \left(\frac{\xi_1(t)}{1+\xi_1^2(t)} + 1\right)\sin(\xi_2(t)) \\
\xi_2(t+1) &= \xi_2(t)\cos(\xi_2(t)) + \xi_1(t)\mathrm{e}^{-(\xi_1^2(t)+\xi_2^2(t))/8} \\
&\quad + \frac{u^3(t)}{1+u^2(t)+0.5\cos(\xi_1(t)+\xi_2(t))} \\
y(t) &= \frac{\xi_1(t)}{1+0.5\sin(\xi_2(t))} + \frac{\xi_2(t)}{1+0.5\sin(\xi_1(t))} + \epsilon(t)
\end{aligned} \tag{4.22}$$

We have used a PWARX model of the form

$$y(t+1) \approx \begin{cases} \varphi^T(t)\theta_1^* & \text{if } x(t) \in X_1 \text{ or } \tilde{x}(t) \in \tilde{X}_1 \\ \vdots & \vdots \\ \varphi^T(t)\theta_N^* & \text{if } x(t) \in X_N \text{ or } \tilde{x}(t) \in \tilde{X}_N \end{cases} \tag{4.23}$$

where

$$x(t) = [y(t), y(t-1), y(t-2), u(t), u(t-1), u(t-2)]^T \tag{4.24}$$

$$\varphi(t) = \left[x(t)^T, 1\right]^T \tag{4.25}$$

$$\tilde{x}(t) = [y(t), y(t-1), y(t-2), u(t-1), u(t-2)]^T \tag{4.26}$$

to approximate the nonlinear system, and observed that the data could be fit fairly well using 4 subsystems.

Based on the identified PWARX model of the nonlinear system, we then design the weighted one-step-ahead control law (4.10) to control the nonlinear system. The nonlinear system is required to track the reference signal

$$r(t) = \sin(2\pi t/20) + \sin(2\pi t/100) \tag{4.27}$$

The simulation result for the control performance using Method I is shown in Fig. 4.6. As can be seen, the control performance is very poor for Method I. The assumption that $\sigma(t) = \sigma(t-1)$ is not valid, leading to the wrong choice of subsystem and thus the wrong computation of control signal.

We next proceed to use Method IV to control the nonlinear system. The simulation result using Method IV is shown in Fig. 4.7. As can be observed, the control performance, although not yet perfect, improved significantly as compared to Method I. However, the computational burden is rather high using Method IV, since a nonlinear optimization problem needs to be solved at each time step. For instance, the simulation time was more than three minutes for the case of having 4 subsystems in the PWARX model.

Finally, we tried out Method V to control the nonlinear system, and the simulation results are shown in Fig. 4.8. As can be seen, the tracking performance is comparable to that of Method IV, but the biggest advantage of Method V is that computation time is very short - only two seconds was needed!

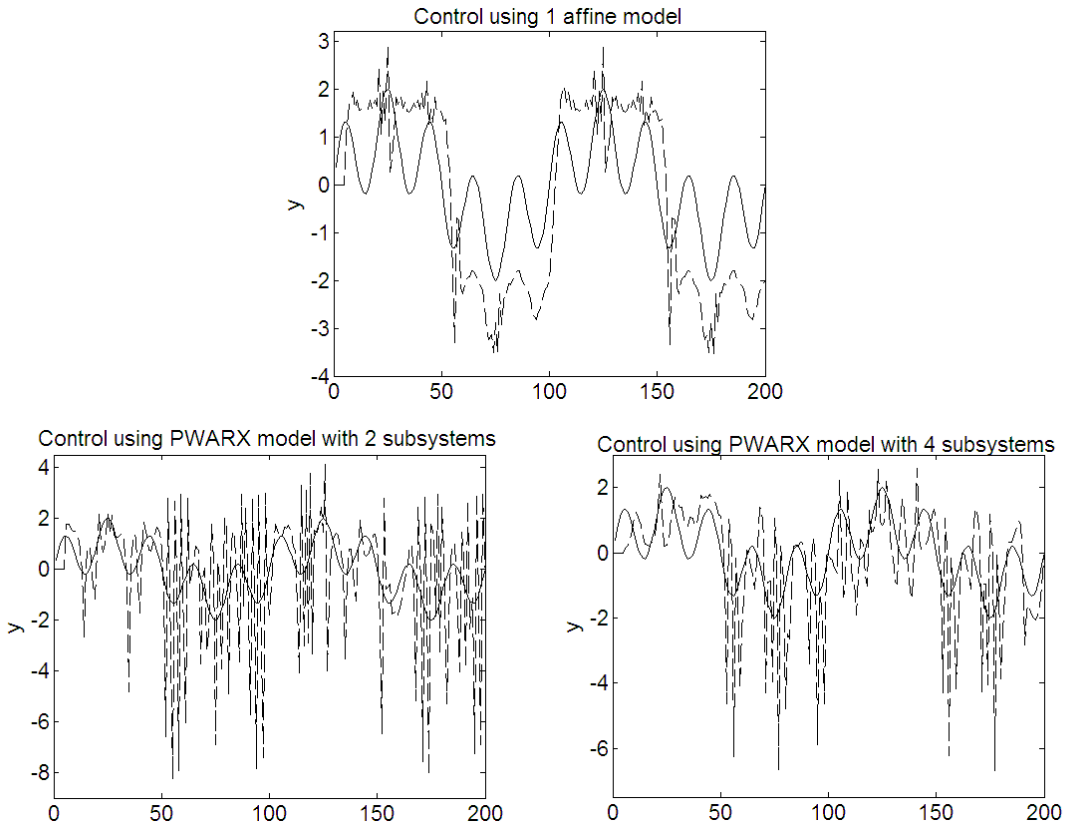

Figure 4.6: Control of the nonlinear system 3 - Method I.
Solid: Reference, dashed: Output of nonlinear system.

4.5 Experimental Studies

4.5.1 Single-Link Robotic Arm

Consider again the single-link robotic arm, whose input-output relationships has been identified using PWARX model in section 3.7.2. There, we showed that the identification error decreases as we have more subsystems in the PWARX model.

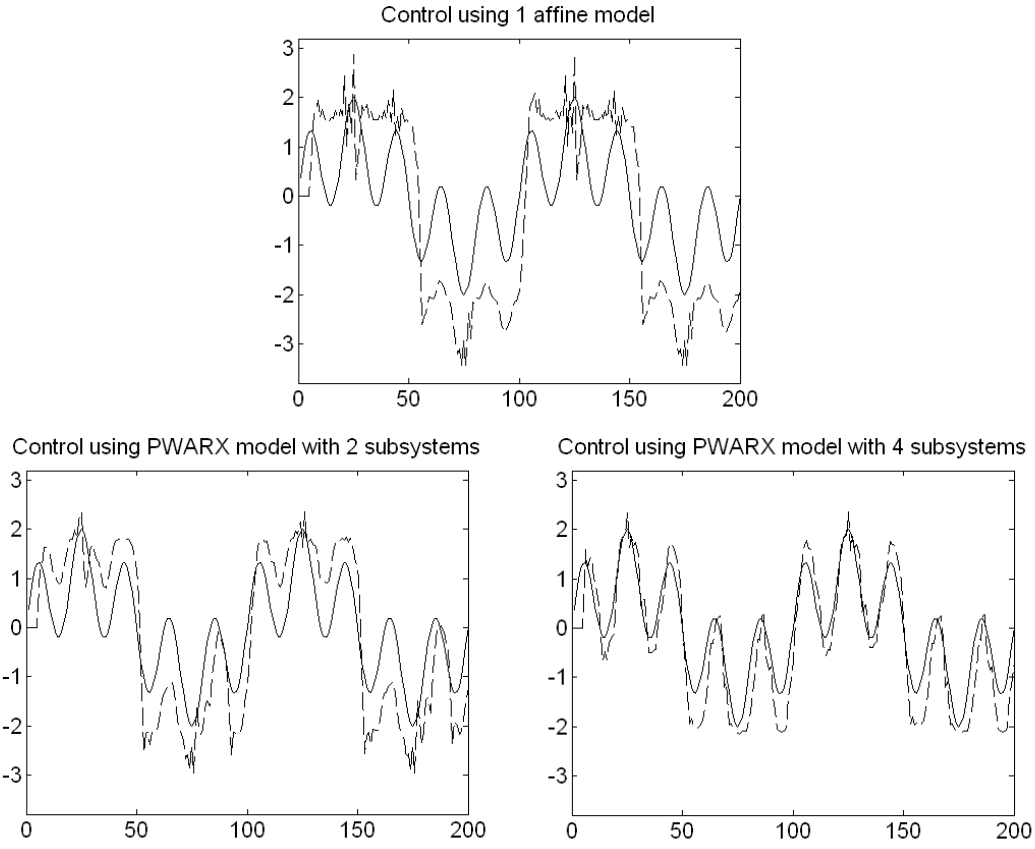

Figure 4.7: Control of the nonlinear system 3 - Method IV.

Solid: Reference, dashed: Output of nonlinear system.

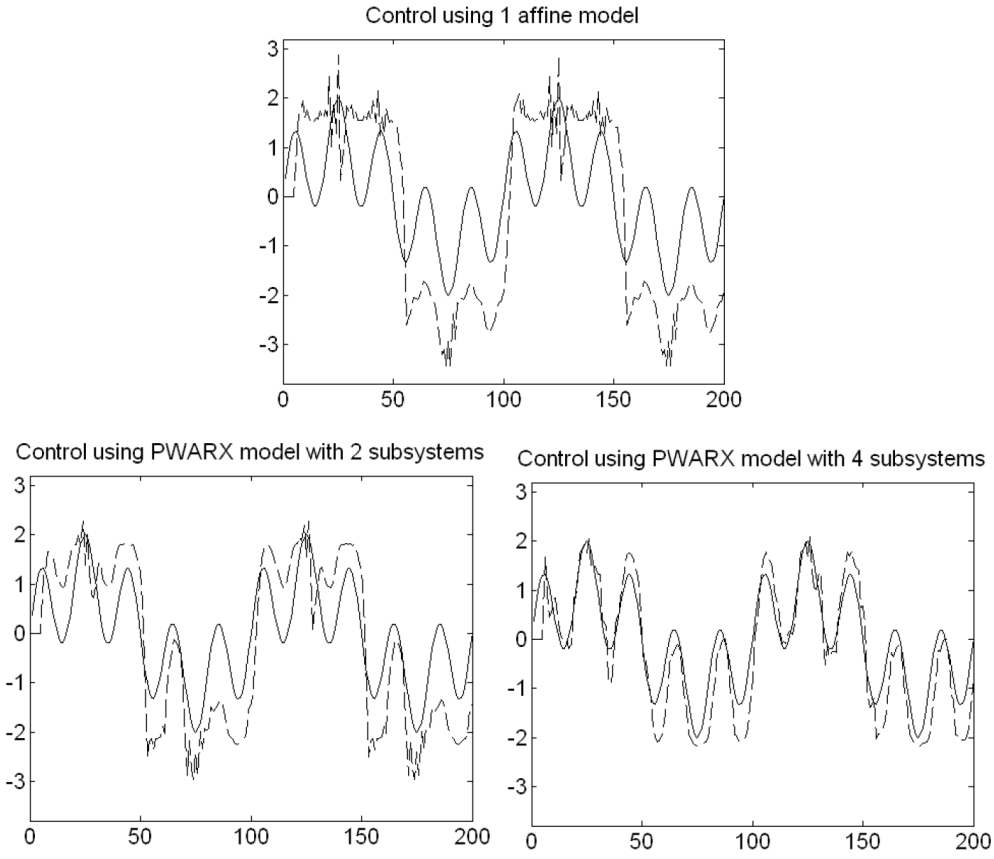

Figure 4.8: Control of the nonlinear system 3 - Method V.

Solid: Reference, dashed: Output of nonlinear system.

Table 4.1: Variance of Tracking Error

No. subsystems	1	2	4	8	PID
Reference Signal 1	0.0155	0.0074	0.0040	0.0032	0.0148
Reference Signal 2	0.0112	0.0059	0.0031	0.0026	0.0152

With an accurate model at hand, we now proceed to control the robotic arm for (angular) position tracking. Recall that we directly used classifier II to estimate the partition of the regressor space, and thus we will apply Method V for the control. Here, we defined two reference signals:

$$r_1(t) = 1.5\sin(2\pi 0.05t) + \pi \tag{4.28}$$

$$r_2(t) = \sin(2\pi 0.03t) + 0.8\sin(2\pi 0.1t + \pi/3) + \pi \tag{4.29}$$

The experimental results are shown in Figs. 4.9 and 4.10 respectively. It can be observed that the tracking errors decrease as the number of subsystems increases. This improvement is also numerically verified as shown through the variance of errors in TABLE 4.1.

A PID-type controller is also designed for comparison purposes. The parameters of the controller are properly tuned to achieve a good tracking performance. The experimental results are shown in Fig. 4.11. It is clear that the tracking accuracy of using our proposed weighted one-step-ahead controller with PWARX model is higher than by using the PID controller.

Remark 4.2 *We wish to point out that in all the simulation and experimental studies, we did not give the explicit equations for the partitioning of regressor space, because they are not a prerequisite for our method to work. We only need the data classifier - which*

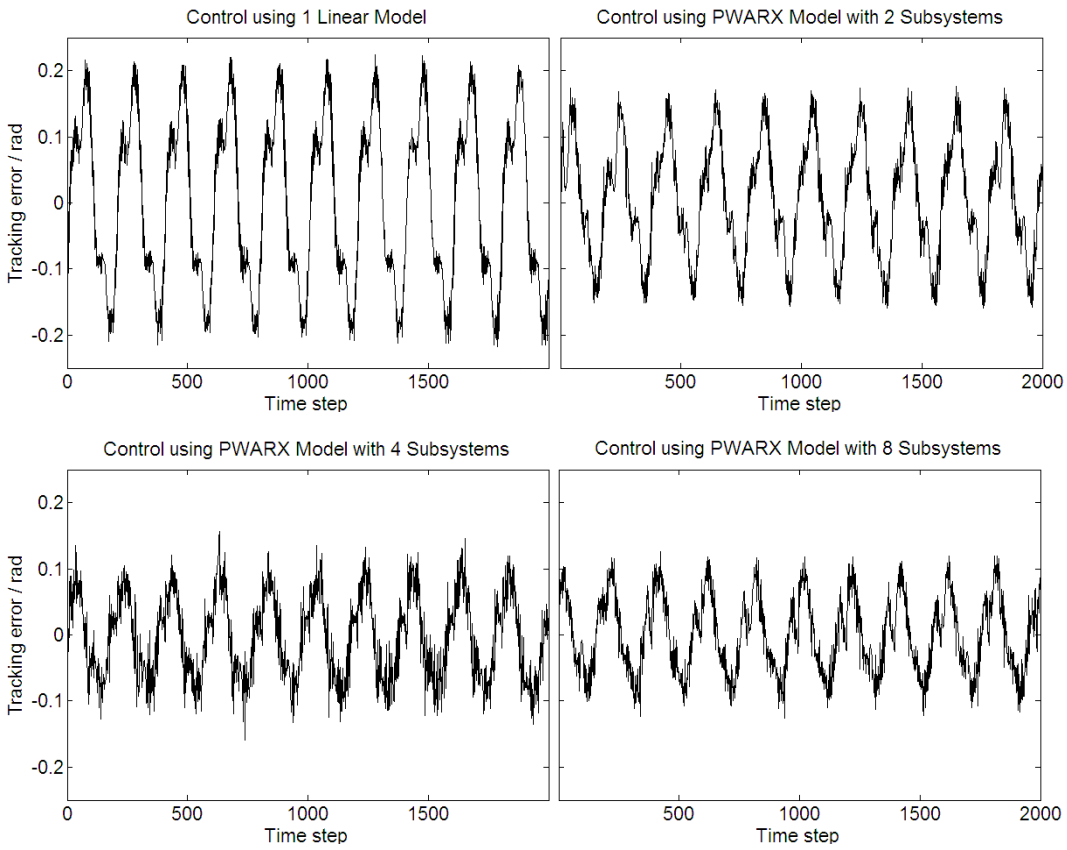

Figure 4.9: Tracking error of the single-link robotic arm, reference signal 1

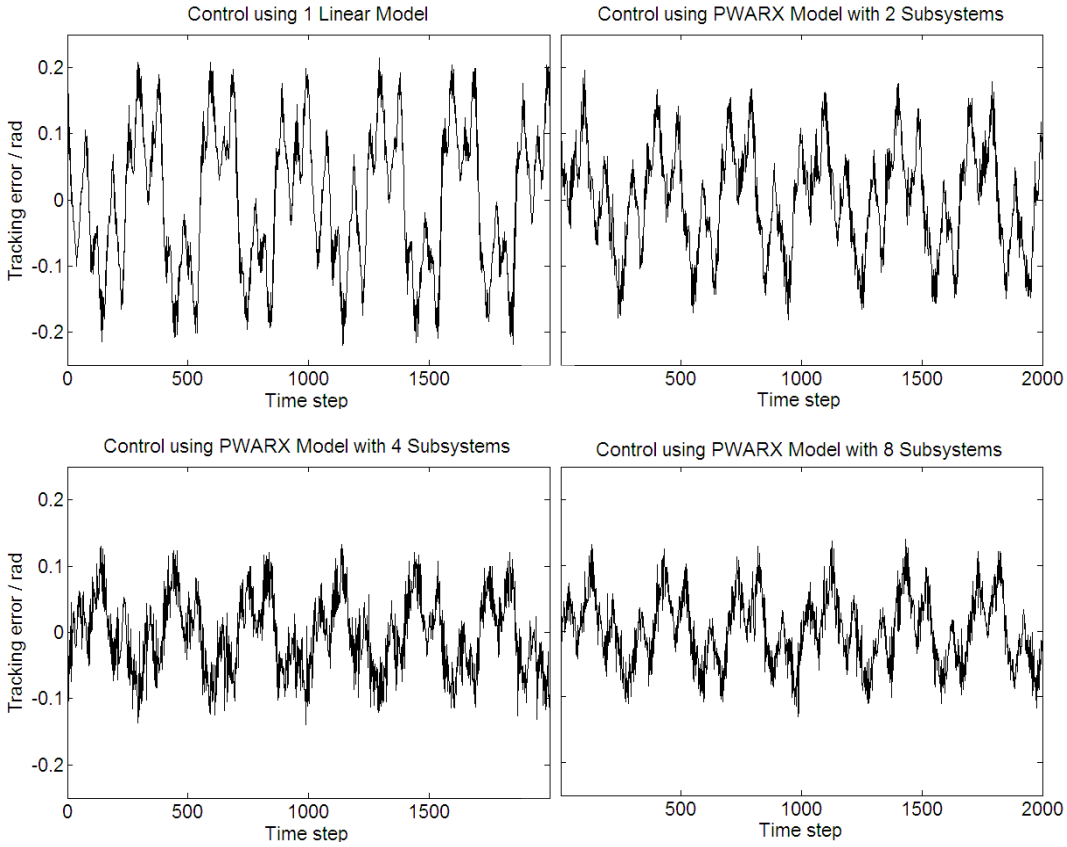

Figure 4.10: Tracking error of the single-link robotic arm, reference signal 2

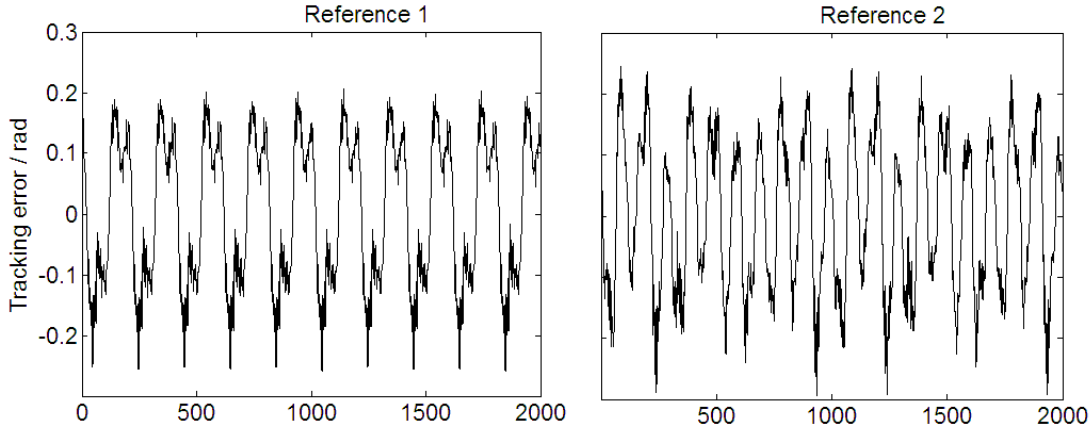

Figure 4.11: Tracking error of the single-link robotic arm using PID control

is a black box itself - to determine the active subsystem, given a particular regression vector.

4.6 Conclusion

In this chapter, we derived weighted one-step-ahead controllers for the piecewise affine models of nonlinear systems, with special attention given to how to determine the active subsystem. This is an important issue because unless we can calculate the control signal based on the model of the correct subsystem, the tracking performance will be very poor. The most prominent novelty in this part of the work is using the modified regressor space for determining the active subsystem, because this method thoroughly bypasses a difficult causality issue. Both simulation and experimental studies show that our proposed control algorithm works very well.

Chapter 5

Input-Output Transition Models for Discrete-Time Switched Linear and Nonlinear Systems

5.1 Introduction

Before going further into this chapter, we would like to first point out that the upcoming results are valid not only for piecewise affine systems, but also for the more general switching systems. As such, we shall use the term switched systems or switching systems more frequently in this chapter.

The problem to be investigated in this chapter is as follows. In the earlier part of this thesis, the number of subsystems of a switching system, N, is assumed to be known, or was determined via a trial-and-error process by observing the wellness-of-fit for both training and test data. Now, assume that we are indeed certain that a switching system consists of N subsystems, where each of these is in the state space form. Would it still be sufficient to use N input-output models to describe the characteristics of the overall switching system? The answer is no! We will show in this chapter that if the N state-space subsystems are each of order p, then there exists a total of N^p input-

output subsystems. As such, it is necessary to define N^p input-output subsystems for an accurate identification and control of the switching system.

The above problem is worthwhile to be investigated, because switched systems are often written in the state space form (see for e.g. [10, 39–44, 47, 49, 56, 57, 70, 71, 74, 75, 82, 85–87]). This is due to the fact that state space models give a more complete description of the system's dynamics, as compared to input-output models. However, in practice, we usually make use of input-output models for system identification or control purposes. Therefore, we need to link the state space models of switched systems appropriately with the input-output models.

In this chapter, we consider switched systems where each of the subsystems is represented by a discrete-time state space model. If we consider each of these state space subsystems individually without switching, we can easily derive the corresponding input-output model, for e.g. by using the well-known z-transform ($C(zI - A)^{-1}B$). However, if switching is taken into account, then none of these input-output models can describe the system's behavior properly at the switching instants, because the output is now influenced by a mixture of new and old system parameters. This leads to inaccurate identification and poor control performance of switched systems during switching instants, as pointed out in the literature, see for e.g. [19, 34].

Although the above problem is known to exist, there are very few effective and readily-usable results (to our best knowledge) that exists in the open literature on the explicit form of the input-output models at the switching instants. For switched linear systems, if we view the discrete-time switched system as a special case of discrete-time time-varying system, then two methods provided in [32] and [33] can be used to derive the input-output models from the Single-Input-Single-Output (SISO) and Multiple-Input-

Single-Output (MISO) time-varying state space models. The method in [33] relies on transforming the state space models into the observable canonical form at every time step, whose time-varying coefficients will then match with the coefficients of the input-output models. However, it is unclear how to utilize the method for more general Multiple-Input-Multiple-Output (MIMO) systems. Similarly, the algorithm in [32] which relies on the "generalized z-Transform" is also difficult to be applied to MIMO systems. Another disadvantage of [32] and [33] is that they are not readily extendible to switched nonlinear systems.

In this chapter, we rigorously develop a rather simple, yet effective, procedure to derive the input-output models from switching state space models. We call the additional input-output models during switching as the "transition models". We further prove that the models are invariant to coordinate transformations of the states. The advantage of our approach is its relative simplicity (and thus an easily adoptable methodology), and its ready applicability for the typically more difficult classes of switched nonlinear systems and MIMO systems.

This chapter is organized as follows. In Section 5.2, we give the mathematical preliminary of deriving the input-output model from a single state space model. We then introduce the concept of transition models for the simple case of a switched linear system with two second order subsystems having the observable canonical form in Section 5.3. Section 5.4 discusses the total number of input-output models for a more general multiple-input-multiple-output switched system, as well as the invariance of the transition models when the state space models undergo coordinate transformations. Here, we also show the applicability of the methodology we developed for suitable classes of switched nonlinear systems. The importance of utilizing these transition models for con-

troller design and system identification is shown via simulation studies in Section 5.5. Finally, the afore-mentioned developments will be cogently summarized in section 5.6.

5.2 Mathematical Preliminary

It is essential that the reader be suitably familiar with the essence of the mathematical derivation from the state-space model to input-output model in the single model case, before the specific subtleties that arise in the switched systems can be appreciated. In view of this, we would like to first show a method for deriving input-output models from state space models for the single model (time-invariant) case.

5.2.1 Linear System

Consider the state space model:

$$\xi(t+1) = A\xi(t) + Bu(t) \tag{5.1}$$

$$y(t) = C\xi(t) \tag{5.2}$$

where $\xi(t) \in \mathbf{R^p}$ is the state vector, $u(t) \in \mathbf{R}$ is the control input, and $y(t) \in \mathbf{R}$ is the system output.

If we start observation at time t, and observe over p steps, then we will have the output sequence, written in a matrix form:

$$\begin{bmatrix} y(t) \\ y(t+1) \\ \vdots \\ y(t+p-1) \end{bmatrix} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{p-1} \end{bmatrix} \xi(t) + \begin{bmatrix} 0 \\ CBu(t) \\ \vdots \\ \sum_{i=0}^{p-2} CA^{p-i-2}Bu(t+i) \end{bmatrix} \tag{5.3}$$

Finally, at time $t+p$, we will obtain:

$$y(t+p) = CA^p\xi(t) + \sum_{i=0}^{p-1} CA^{p-i-1}Bu(t+i) \tag{5.4}$$

The task now is to express $\xi(t)$ in (5.4) in terms of input and output. If we assume that the system is observable, i.e. the observability matrix:

$$O = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{p-1} \end{bmatrix} \tag{5.5}$$

is of full rank or invertible, then we can get $\xi(t)$ from equation (5.3) as:

$$\xi(t) = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{p-1} \end{bmatrix}^{-1} \times \begin{bmatrix} y(t) \\ y(t+1) - CBu(t) \\ \vdots \\ y(t+p-1) - \sum_{i=0}^{p-2} CA^{p-i-2}Bu(t+i) \end{bmatrix} \tag{5.6}$$

Finally, we substitute (5.6) into (5.4) and obtain:

$$\begin{aligned} y(t+p) = CA^p & \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{p-1} \end{bmatrix}^{-1} \times \begin{bmatrix} y(t) \\ y(t+1) - CBu(t) \\ \vdots \\ y(t+p-1) - \sum_{i=0}^{p-2} CA^{p-i-2}Bu(t+i) \end{bmatrix} \\ & + \sum_{i=0}^{p-1} CA^{p-i-1}Bu(t+i) \end{aligned} \tag{5.7}$$

Equation (5.7) is the input-output model corresponding to the state space model (5.1) and (5.2).

5.2.2 Nonlinear System

Consider the nonlinear state space model:

$$\xi(t+1) = f(\xi(t), u(t)) \tag{5.8}$$

$$y(t) = h(\xi(t)) \tag{5.9}$$

where $\xi(t) \in \mathbf{R^p}$, $u(t) \in \mathbf{R}$, $y(t) \in \mathbf{R}$. The state transition map, f is a smooth map from $\mathbf{R^{p+1}}$ to $\mathbf{R^p}$, and h is a smooth map from $\mathbf{R^p}$ to $\mathbf{R}$.

Assuming the origin is the equilibrium state under zero input (i.e. $f(0,0) = 0$ and $h(0) = 0$), then the linearization of the system about the origin can be expressed as:

$$\xi(t+1) = A\xi(t) + bu(t) \tag{5.10}$$

$$y(t) = c\xi(t) \tag{5.11}$$

where:

$$A = \frac{\partial f(\xi,u)}{\partial \xi}|_{(0,0)}, b = \frac{\partial f(\xi,u)}{\partial u}|_{(0,0)}, c = \frac{\partial h(\xi)}{\partial \xi}|_{(0)} \tag{5.12}$$

The p-step observation map from state space $\mathbf{R^p}$ to output sequence space $\mathbf{R^p}$ for any input sequence $u(t), \ldots, u(t+p-2)$, is defined as:

$$y(t) = h(\xi(t))$$

$$y(t+1) = h(\xi(t+1)) = h(f(\xi(t), u(t)))$$

$$\vdots$$

$$y(t+p-1) = h(f(\ldots f(f(\xi(t), u(t)), u(t+1)), \ldots, u(t+p-2))) \tag{5.13}$$

which may be rewritten in a more compact form as:

$$Y(t) = H(\xi(t), U_t) \tag{5.14}$$

where:

$$U_t = [u(t), \ldots, u(t+p-2)]^T \in \mathbf{R^{p-1}} \tag{5.15}$$

The Jacobian matrix of H w.r.t. ξ evaluated at the origin is:

$$\frac{\partial H}{\partial \xi}|_{(0,0)} = \begin{bmatrix} c \\ cA \\ \vdots \\ cA^{p-1} \end{bmatrix} \tag{5.16}$$

which is exactly the observability matrix for the linearized system. Hence, if the linearized system is observable, by the implicit function theorem, it follows that in a neighborhood of the origin, the state $\xi(t)$ can be expressed as a function of $Y(t)$ and U_t, i.e.:

$$\xi(t) = \Phi(Y(t), U_t) \tag{5.17}$$

The output at time $t+p$ is:

$$y(t+p) = h(f(\dots f(f(\xi(t), u(t)), u(t+1)) \dots, u(t+p-1))) \tag{5.18}$$

Substituting equation (5.17) into (5.18) leads to:

$$y(t+p) = F(y(t+p-1), \dots, y(t), u(t+p-1), \dots, u(t)) \tag{5.19}$$

Equation (5.19) is the input-output model corresponding to the nonlinear state-space model (5.8) and (5.9) in a neighborhood of the origin. The above procedure was shown in [38]. The extensions of the above result to global input-output representation for general discrete-time nonlinear system can be found in [81].

5.3 Simple Case: Switched Linear System with Two Second Order Subsystems in Observable Canonical Form

In this section, we would like to introduce the concept of transition models with the help of a simple switched linear system with two second order subsystems having the observable canonical form. This is because in this case, the input-output models can be derived graphically and thus understood intuitively.

The system under consideration is:

$$\xi(t+1) = \begin{cases} A_1\xi(t) + B_1u(t) \\ \qquad = \begin{bmatrix} -a_{11} & 1 \\ -a_{12} & 0 \end{bmatrix} \xi(t) + \begin{bmatrix} b_{11} \\ b_{12} \end{bmatrix} u(t) \\ \qquad \text{if subsystem 1} \\ \\ A_2\xi(t) + B_2u(t) \\ \qquad = \begin{bmatrix} -a_{21} & 1 \\ -a_{22} & 0 \end{bmatrix} \xi(t) + \begin{bmatrix} b_{21} \\ b_{22} \end{bmatrix} u(t) \\ \qquad \text{if subsystem 2} \end{cases} \tag{5.20}$$

$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \xi(t) \tag{5.21}$$

We assume that the states do not experience sudden jumps when switching occurs.

If the system always stays in the first subsystem, then the input-output model can be easily derived as:

$$y(t+1) = -a_{11}y(t) - a_{12}y(t-1) + b_{11}u(t) + b_{12}u(t-1). \tag{5.22}$$

Similarly, if the system always stays in the second subsystem, then the corresponding input-output model is:

$$y(t+1) = -a_{21}y(t) - a_{22}y(t-1) + b_{21}u(t) + b_{22}u(t-1). \tag{5.23}$$

However, at switching instants, neither (5.22) nor (5.23) models the system correctly. It will be shown shortly that the input-output models should actually be:

$$y(t+1) = -a_{21}y(t) - a_{12}y(t-1) + b_{21}u(t) + b_{12}u(t-1) \tag{5.24}$$

when switching from subsystem 1 to subsystem 2 at time t, and:

$$y(t+1) = -a_{11}y(t) - a_{22}y(t-1) + b_{11}u(t) + b_{22}u(t-1) \tag{5.25}$$

when switching from subsystem 2 to subsystem 1.

The models (5.24) and (5.25) are what we termed as "transition models" because they are valid only at switching (transition) instants. Notice that these models have a mixture of parameters from the most current and previous subsystems.

Now, we would like to show graphically how to derive these transition models. Imagine that before and at time t, the system is in the first subsystem, $\xi(t+1) = A_1\xi(t) + B_1u(t)$. Shown in Fig. 5.1 is the observable canonical realization of the system with parameters of the first subsystem:

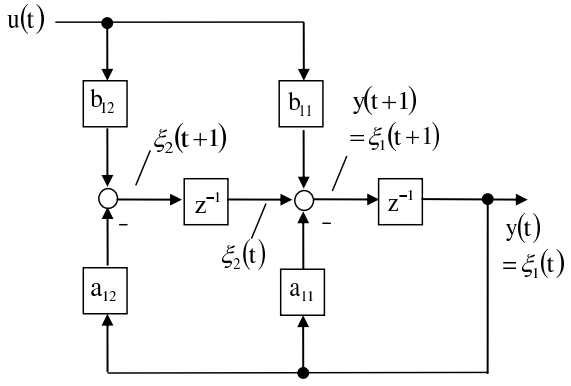

Figure 5.1: Subsystem 1 and its signals

In Fig. 5.1, the labeled signals are as follows:

$$\xi_2(t+1) = b_{12}u(t) - a_{12}y(t) \tag{5.26}$$

$$\xi_2(t) = b_{12}u(t-1) - a_{12}y(t-1) \tag{5.27}$$

$$\begin{aligned} y(t+1) &= b_{11}u(t) - a_{11}y(t) + \xi_2(t) \\ &= b_{11}u(t) - a_{11}y(t) + b_{12}u(t-1) - a_{12}y(t-1) \end{aligned} \tag{5.28}$$

It can be seen that the signal $y(t+1)$ in (5.28) equals the model given by (5.22).

Next, assume that at the next time step (i.e. $t+1$), the system switches to subsystem 2, such that $\xi(t+2) = A_2\xi(t+1) + B_2u(t+1)$. The parameters and the signals of the system are shown in Fig. 5.2.

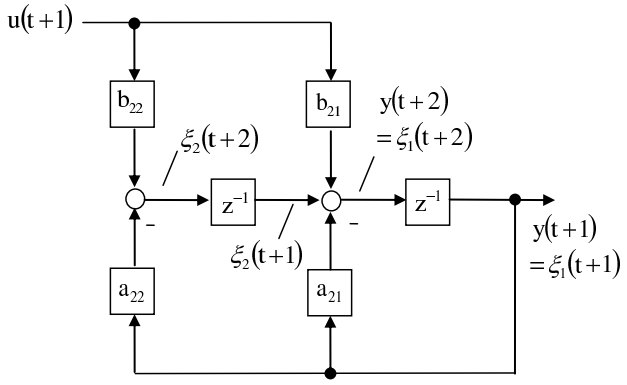

Figure 5.2: Signals of the system when switching to subsystem 2

In Fig. 5.2, the predicted output $y(t+2)$ is:

$$y(t+2) = b_{21}u(t+1) - a_{21}y(t+1) + \xi_2(t+1) \tag{5.29}$$

Substituting (5.26) into (5.29), we obtain:

$$y(t+2) = b_{21}u(t+1) - a_{21}y(t+1) + b_{12}u(t) - a_{12}y(t) \tag{5.30}$$

The model (5.30) is exactly the one given in (5.24). Notice that the prediction of $y(t+2)$ is affected by parameters of both the previous and the current subsystems at the switching instant, $t+1$. We would like to emphasize that the model (5.30) is only valid at the switching instant $t+1$ to predict the output at the next step, $t+2$. The input-output model will be the normal one (5.23) if the system stays at system 2 thereafter.

Following the same idea, one can easily show that (5.25) correctly models the system when there is a switch from subsystem 2 to subsystem 1 at time t.

5.4 Main Result

The derivation of the transition models via the graphical method is only suitable for switched linear systems in the observable canonical form, because of the straightforward relationship between the states, input and output. In this section, we will show the

mathematical derivation of the input-output models for general, possibly multiple-input-multiple-output state space models.

5.4.1 Switched Linear Systems

The system under consideration is:

$$\xi(t+1) = A_{\sigma(t)}\xi(t) + B_{\sigma(t)}u(t) \tag{5.31}$$

$$y(t) = C\xi(t) \tag{5.32}$$

where $\xi(t) \in \mathbf{R^p}$ is the state vector, $u(t) \in \mathbf{R^q}$ is the control input, and $y(t) \in \mathbf{R^r}$ is the system output. The matrices $A_{\sigma(t)}$, $B_{\sigma(t)}$ and C are appropriately dimensioned real matrices, whereas $\sigma(t) \in \{1, 2, \ldots, N\}$ is the switching signal. Note that $\sigma(t) \in \{1, 2, \ldots, N\}$ means that the switched system consists of N subsystems. We also constrain the matrix C to be time-invariant, which implies that the same type of output signals is measured for all subsystems. Furthermore, we assume that the states do not jump during switching.

In order to derive the explicit form of the input-output models, we need to impose some condition regarding the observability of the system.

Assumption 5.1 *The system (5.31), (5.32) is uniformly observable, i.e. the observability matrix [33]:*

$$O(t) = \begin{bmatrix} C \\ CA_{\sigma(t)} \\ \vdots \\ CA_{\sigma(t+p-2)} \ldots A_{\sigma(t)} \end{bmatrix} \tag{5.33}$$

has full rank for all t.

Now, we will derive the input-output models using the procedure as shown in Section 5.2. If we start observation at time t and observe over p steps, then we will have the

output sequence:

$$\begin{bmatrix} y(t) \\ y(t+1) \\ \vdots \\ y(t+p-1) \end{bmatrix} = \begin{bmatrix} C \\ CA_{\sigma(t)} \\ \vdots \\ CA_{\sigma(t+p-2)} \dots A_{\sigma(t)} \end{bmatrix} \xi(t) + \begin{bmatrix} 0 \\ CB_{\sigma(t)}u(t) \\ \vdots \\ \gamma \end{bmatrix} \tag{5.34}$$

where:

$$\begin{aligned} \gamma = C[&A_{\sigma(t+p-2)} \dots A_{\sigma(t+1)} B_{\sigma(t)} u(t) \\ &+ A_{\sigma(t+p-2)} \dots A_{\sigma(t+2)} B_{\sigma(t+1)} u(t+1) \\ &+ \dots + B_{\sigma(t+p-2)} u(t+p-2)] \end{aligned} \tag{5.35}$$

At time $t+p$, we will have:

$$\begin{aligned} y(t+p) = &CA_{\sigma(t+p-1)} \dots A_{\sigma(t)} \xi(t) \\ &+ CA_{\sigma(t+p-1)} \dots A_{\sigma(t+1)} B_{\sigma(t)} u(t) \\ &+ CA_{\sigma(t+p-1)} \dots A_{\sigma(t+2)} B_{\sigma(t+1)} u(t+1) \\ &+ \dots + CB_{\sigma(t+p-1)} u(t+p-1) \end{aligned} \tag{5.36}$$

The task now is to express $\xi(t)$ in (5.36) in terms of input and output. By assumption 5.1, we can express $\xi(t)$ from (5.34) as:

$$\begin{aligned} \xi(t) &= \begin{bmatrix} C \\ CA_{\sigma(t)} \\ \vdots \\ CA_{\sigma(t+p-2)} \dots A_{\sigma(t)} \end{bmatrix}^{+} \times \begin{bmatrix} y(t) \\ y(t+1) - CB_{\sigma(t)}u(t) \\ \vdots \\ y(t+p-1) - \gamma \end{bmatrix} \\ &= O^{+}(t) \begin{bmatrix} y(t) \\ y(t+1) - CB_{\sigma(t)}u(t) \\ \vdots \\ y(t+p-1) - \gamma \end{bmatrix} \end{aligned} \tag{5.37}$$

where $O^{+}(t)$ denotes the pseudo-inverse of $O(t)$:

$$O^{+}(t) = (O^{T}(t)O(t))^{-1}O^{T}(t) \tag{5.38}$$

Substituting (5.37) into (5.36), we will obtain the input-output model:

$$\begin{aligned} y(t+p) &= CA_{\sigma(t+p-1)}\dots A_{\sigma(t)} \\ &\quad \times \begin{bmatrix} C \\ CA_{\sigma(t)} \\ \vdots \\ CA_{\sigma(t+p-2)}\dots A_{\sigma(t)} \end{bmatrix}^{+} \times \begin{bmatrix} y(t) \\ y(t+1) - CB_{\sigma(t)}u(t) \\ \vdots \\ y(t+p-1) - \gamma \end{bmatrix} \\ &\quad + CA_{\sigma(t+p-1)}\dots A_{\sigma(t+1)}B_{\sigma(t)}u(t) \\ &\quad + CA_{\sigma(t+p-1)}\dots A_{\sigma(t+2)}B_{\sigma(t+1)}u(t+1) \\ &\quad + \dots + CB_{\sigma(t+p-1)}u(t+p-1) \end{aligned} \tag{5.39}$$

which covers all the cases of either staying in one subsystem or switching among the subsystems, depending on the switching sequence $\{\sigma(t), \sigma(t+1), \dots, \sigma(t+p-1)\}$.

Now we are ready to state the following theorem which is the first main result of this chapter.

Theorem 5.1 *Consider the general multiple-input-multiple-output state space model (5.31), (5.32). Subject to assumption 5.1, i.e. the uniform observability condition, a total of N^p input-output models is required to perfectly describe the system behavior for all possible switching sequences. Furthermore, these input-output models are invariant under state transformations.*

Proof: We first prove the total number of input-output models. We see from (5.39) that $y(t+p)$ depends on the switching sequence $\{\sigma(t), \sigma(t+1), \dots, \sigma(t+p-1)\}$, where $\sigma(i) \in \{1, 2, \dots, N\}$. Thus, depending on the switching sequence, one can get a total of:

$$\underbrace{N}_{t} \times \underbrace{N}_{t+1} \times \dots \times \underbrace{N}_{t+p-1} = N^p \tag{5.40}$$

different combinations of $\{\sigma(t), \sigma(t+1), \dots, \sigma(t+p-1)\}$. This will result in N^p different input-output models to cover all the cases where the system stays in one subsystem or switches.

Next, we would like to prove the invariance of the transition models when the state models undergo a coordinate transformation. Consider the following switched system:

$$\bar{\xi}(t+1) = \bar{A}_{\sigma(t)}\bar{\xi}(t) + \bar{B}_{\sigma(t)}u(t) \tag{5.41}$$

$$y(t) = \bar{C}\bar{\xi}(t) \tag{5.42}$$

where:

$$\bar{\xi}(t) = T\xi(t) \tag{5.43}$$

$$\bar{A}_{\sigma(t)} = TA_{\sigma(t)}T^{-1} \tag{5.44}$$

$$\bar{B}_{\sigma(t)} = TB_{\sigma(t)} \tag{5.45}$$

$$\bar{C} = CT^{-1} \tag{5.46}$$

Following the procedure shown earlier, we will obtain:

$$\begin{aligned} y(t+p) &= \bar{C}\bar{A}_{\sigma(t+p-1)}\dots\bar{A}_{\sigma(t)} \\ &\times \begin{bmatrix} \bar{C} \\ \bar{C}\bar{A}_{\sigma(t)} \\ \vdots \\ \bar{C}\bar{A}_{\sigma(t+p-2)}\dots\bar{A}_{\sigma(t)} \end{bmatrix}^{+} \times \begin{bmatrix} y(t) \\ y(t+1) - \bar{C}\bar{B}_{\sigma(t)}u(t) \\ \vdots \\ y(t+p-1) - \bar{\gamma} \end{bmatrix} \\ &+ \bar{C}\bar{A}_{\sigma(t+p-1)}\dots\bar{A}_{\sigma(t+1)}\bar{B}_{\sigma(t)}u(t) \\ &+ \bar{C}\bar{A}_{\sigma(t+p-1)}\dots\bar{A}_{\sigma(t+2)}\bar{B}_{\sigma(t+1)}u(t+1) \\ &+ \dots + \bar{C}\bar{B}_{\sigma(t+p-1)}u(t+p-1) \end{aligned} \tag{5.47}$$

where:

$$\begin{aligned} \bar{\gamma} = \bar{C}[&\bar{A}_{\sigma(t+p-2)}\dots\bar{A}_{\sigma(t+1)}\bar{B}_{\sigma(t)}u(t) \\ &+ \bar{A}_{\sigma(t+p-2)}\dots\bar{A}_{\sigma(t+2)}\bar{B}_{\sigma(t+1)}u(t+1) \\ &+ \dots + \bar{B}_{\sigma(t+p-2)}u(t+p-2)] \end{aligned} \tag{5.48}$$

Now, substitute equations (5.44), (5.45) and (5.46) into (5.47) and (5.48). After some simple cancellation of T with T^{-1}, we will arrive at:

$$\begin{aligned} y(t+p) &= CA_{\sigma(t+p-1)}\dots A_{\sigma(t)}T^{-1} \\ &\quad \times \begin{bmatrix} CT^{-1} \\ CA_{\sigma(t)}T^{-1} \\ \vdots \\ CA_{\sigma(t+p-2)}\dots A_{\sigma(t)}T^{-1} \end{bmatrix}^{+} \times \begin{bmatrix} y(t) \\ y(t+1) - CB_{\sigma(t)}u(t) \\ \vdots \\ y(t+p-1) - \gamma \end{bmatrix} \\ &\quad + CA_{\sigma(t+p-1)}\dots A_{\sigma(t+1)}B_{\sigma(t)}u(t) \\ &\quad + CA_{\sigma(t+p-1)}\dots A_{\sigma(t+2)}B_{\sigma(t+1)}u(t+1) \\ &\quad + \dots + CB_{\sigma(t+p-1)}u(t+p-1) \end{aligned} \tag{5.49}$$

How do we deal with the remaining T^{-1} in (5.49)? It is observed that:

$$\begin{aligned} T^{-1}\begin{bmatrix} CT^{-1} \\ CA_{\sigma(t)}T^{-1} \\ \vdots \\ CA_{\sigma(t+p-2)}\dots A_{\sigma(t)}T^{-1} \end{bmatrix}^{+} &= T^{-1}\left[\begin{bmatrix} C \\ CA_{\sigma(t)} \\ \vdots \\ CA_{\sigma(t+p-2)}\dots A_{\sigma(t)} \end{bmatrix} T^{-1}\right]^{+} \\ &= \underbrace{T^{-1}T}_{I}\begin{bmatrix} C \\ CA_{\sigma(t)} \\ \vdots \\ CA_{\sigma(t+p-2)}\dots A_{\sigma(t)} \end{bmatrix}^{+} \end{aligned} \tag{5.50}$$

Substituting (5.50) into (5.49), we get back the same input-output models as if they were derived using the original coordinate. This shows that the input-output models, including the transition models, are invariant under coordinate transformation of the state models. This completes the proof for Theorem 5.1.

5.4.2 Switched Nonlinear Systems

In this subsection, we would like to show the existence of the input-output transition models for switched nonlinear systems, at least locally in a neighborhood of the origin.

To make the derivation more transparent, we will confine our attention to SISO systems subsequently. But note that all the results can be readily extended to MIMO systems as well.

Consider the following switched nonlinear system:

$$\xi(t+1) = f_{\sigma(t)}(\xi(t), u(t)) \tag{5.51}$$

$$y(t) = h(\xi(t)) \tag{5.52}$$

where $\xi(t) \in \mathbf{R^p}$, $u(t) \in \mathbf{R}$, $y(t) \in \mathbf{R}$, $f_{\sigma(t)}$ are smooth state transition maps from $\mathbf{R^{p+1}}$ to $\mathbf{R^p}$, and h is a smooth map from $\mathbf{R^p}$ to $\mathbf{R}$. $\sigma(t) \in \{1, 2, \ldots, N\}$ is the switching signal. Note that $\sigma(t) \in \{1, 2, \ldots, N\}$ means that the switched system consists of N subsystems. We also constrain the function h to be time-invariant, which implies that the same type of output signals is measured for all subsystems. Furthermore, we assume that the states do not experience sudden jumps when switching happens.

Assuming the origin is the equilibrium state under zero input (i.e. $f_{\sigma(t)}(0,0) = 0$ and $h(0) = 0$), then the linearization of the system about the origin can be expressed as:

$$\xi(t+1) = A_{\sigma(t)}\xi(t) + b_{\sigma(t)}u(t) \tag{5.53}$$

$$y(t) = c\xi(t) \tag{5.54}$$

where:

$$A_{\sigma(t)} = \frac{\partial f_{\sigma(t)}(\xi, u)}{\partial \xi}|_{(0,0)} \tag{5.55}$$

$$b_{\sigma(t)} = \frac{\partial f_{\sigma(t)}(\xi, u)}{\partial u}|_{(0,0)} \tag{5.56}$$

$$c = \frac{\partial h(\xi)}{\partial \xi}|_{(0)} \tag{5.57}$$

Assumption 5.2 *The linearized system (5.53) and (5.54) is uniformly observable, i.e. the observability matrix [33]:*

$$O(t) = \begin{bmatrix} c \\ cA_{\sigma(t)} \\ \vdots \\ cA_{\sigma(t+p-2)} \dots A_{\sigma(t)} \end{bmatrix} \tag{5.58}$$

has full rank for all t.

The p-step observation map from state space $\mathbf{R^p}$ to output sequence space $\mathbf{R^p}$ for any input sequence $u(t), \dots, u(t+p-2)$ is defined as:

$$y(t) = h(\xi(t))$$

$$y(t+1) = h(\xi(t+1)) = h(f_{\sigma(t)}(\xi(t), u(t)))$$

$$\vdots$$

$$y(t+p-1) = h(f_{\sigma(t+p-2)}(\dots f_{\sigma(t+1)}(f_{\sigma(t)}(\xi(t), u(t)), u(t+1)), \dots, u(t+p-2))) \tag{5.59}$$

which may be rewritten in a more compact form as:

$$Y(t) = H(\xi(t), U_t) \tag{5.60}$$

where:

$$U_t = [u(t), \dots, u(t+p-2)]^T \in \mathbf{R^{p-1}} \tag{5.61}$$

The Jacobian matrix of H with respect to ξ evaluated at the origin is:

$$\frac{\partial H}{\partial \xi}|_{(0,0)} = \begin{bmatrix} c \\ cA_{\sigma(t)} \\ \vdots \\ cA_{\sigma(t+p-2)} \dots A_{\sigma(t+1)} A_{\sigma(t)} \end{bmatrix} \tag{5.62}$$

which is exactly the observability matrix for the linearized system. Hence, since the linearized system is assumed to be uniformly observable (assumption 5.2), by the implicit

function theorem, it follows that in a neighborhood of the origin, the state $\xi(t)$ can be expressed as a function of $Y(t)$ and U_t, i.e.:

$$\xi(t) = \Phi(Y(t), U_t) \tag{5.63}$$

Considering the output at time $t+p$:

$$y(t+p) = h(f_{\sigma(t+p-1)}(\dots f_{\sigma(t+1)}(f_{\sigma(t)}(\xi(t), u(t)), u(t+1)), \dots, u(t+p-1))) \tag{5.64}$$

Substituting (5.63) into (5.64) leads to:

$$y(t+p) = F_{\sigma(t+p-1)\dots\sigma(t)}(y(t), \dots, y(t+p-1), u(t), \dots, u(t+p-1)) \tag{5.65}$$

Equation (5.65) defines the input-output models for the switched nonlinear system (5.51), (5.52). Note that the function $F_{\sigma(t+p-1)\sigma(t+p-2)\dots\sigma(t+1)\sigma(t)}$ depends on the switching sequence, thus confirming our claim that transition models do exist for switched nonlinear systems.

Now we are ready to state the following theorem which is the second main result of this chapter.

Theorem 5.2 *Consider the switched nonlinear system (5.51), (5.52). Subject to assumption 5.2, i.e. the uniform observability condition for the linearized system (5.53), (5.54), then a total of N^p input-output models is required to perfectly describe the system behavior in the neighborhood of the origin for all possible switching sequences.*

Proof: We see from (5.65) that $y(t+p)$ depends on the switching sequence $\{\sigma(t), \dots, \sigma(t+p-1)\}$, where $\sigma(i) \in \{1, 2, \dots, N\}$. Thus, depending on the switching sequence, one can get a total of:

$$\underbrace{N}_{t} \times \underbrace{N}_{t+1} \times \cdots \times \underbrace{N}_{t+p-1} = N^p \tag{5.66}$$

different combinations of $\{\sigma(t), \sigma(t+1), \ldots, \sigma(t+p-1)\}$. This will result in N^p different input-output models to cover all the cases where the system stays in one subsystem or switches. This completes the proof for Theorem 5.2.

5.5 Simulation Studies

5.5.1 Design of One-Step-Ahead Controllers for Switched Linear System

Having derived the transition models, we would like to utilize them to design controller to achieve good tracking performance of switched systems. Here, the controller we design is the simple one-step-ahead controller.

Consider the same system as shown in (5.20) and (5.21), with $a_{11} = -0.7859$, $a_{12} = 0.3679$, $b_{11} = 0.3403$, $b_{12} = 0.2417$, $a_{21} = 0.3630$, $a_{22} = 0.3012$, $b_{12} = 1.0186$ and $b_{22} = 0.6456$. Without taking into account the transition models, we would only design two controllers for the input-output models (5.22) and (5.23) as follows: at time t, if the system is in subsystem 1, use:

$$u(t) = \frac{y^*(t+1) + a_{11}y(t) + a_{12}y(t-1) - b_{12}u(t-1)}{b_{11}} \tag{5.67}$$

and if the system is in subsystem 2, use:

$$u(t) = \frac{y^*(t+1) + a_{21}y(t) + a_{22}y(t-1) - b_{22}u(t-1)}{b_{21}} \tag{5.68}$$

where $y^*(t+1)$ denotes the one-step-ahead desired output, which is a square wave in our simulation.

Now, if there is frequent switching between the two subsystems with an equal probability of either subsystem being active at each time step, then using the controllers above results in poor tracking performance, as can be seen from Fig. 5.3(a).

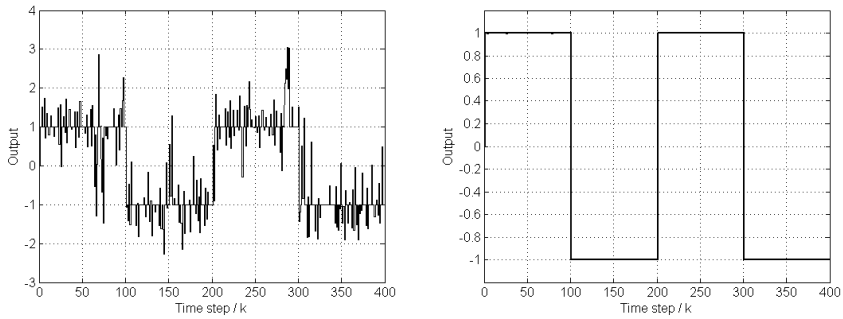

Figure 5.3: Tracking performance by using (a) two controllers only (b) four controllers

However, with the knowledge of the transition models (5.24) and (5.25), we can now design two additional controllers as follows. Assuming the switching signal is known, at the instant of switching from subsystem 1 to subsystem 2, use:

$$u(t) = \frac{y^*(t+1) + a_{21}y(t) + a_{12}y(t-1) - b_{12}u(t-1)}{b_{21}} \tag{5.69}$$

and when switching from subsystem 2 to subsystem 1, use:

$$u(t) = \frac{y^*(t+1) + a_{11}y(t) + a_{22}y(t-1) - b_{22}u(t-1)}{b_{11}} \tag{5.70}$$

With these two additional controllers, the tracking performance is improved significantly despite frequent switching between the subsystems, as can be observed from Fig. 5.3(b).

To check that the controllers can still work well under the influence of measurement noise, a white noise of variance 0.0011 (equivalent to standard deviation $\sigma = 0.0333$ or 3σ value of 0.1) is added to the output measurement. The performance of the four controllers, which take the transition models into account, is still much better than that by using only two controllers, as can be observed from Fig. 5.4.

The importance of including the transition models in the controller design for switched system is clearly seen from the simple examples above.

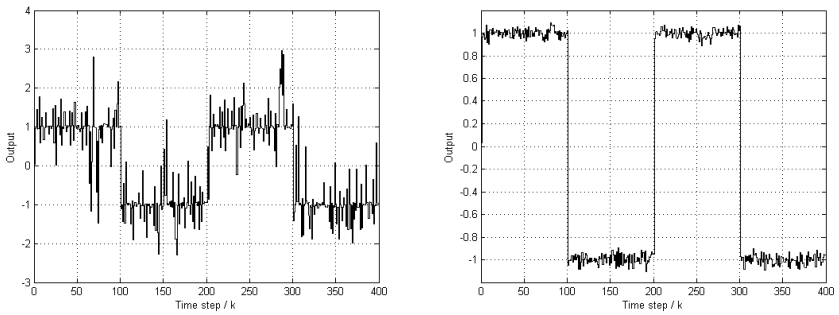

Figure 5.4: Tracking performance by using (a) two controllers only (b) four controllers when output measurement is noisy

5.5.2 Identification of Switched Linear Systems using Multiple Models

The multiple model based identification is a popular method to identify switched systems. The scheme is shown in Fig. 5.5. One fixes N identification models, where N is the expected number of input-output models. "Logic" then decides which model shall be updated according to the switching signal.

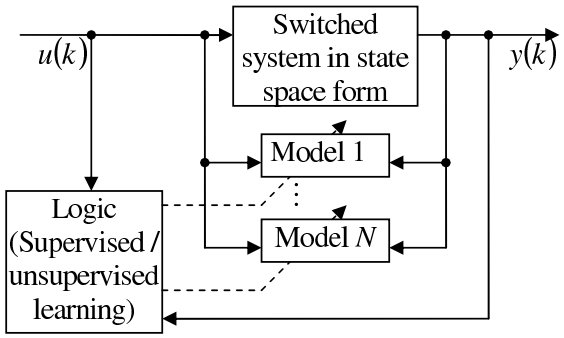

Figure 5.5: Identification of switched systems using multiple models

From our discussion thus far, it is easy to understand that if we do not define enough input-output models to cover the transition models, then the "logic" cannot decide unambiguously which model to update during the switching instants. This is essentially the problem mentioned in [19, 34]. If we now include the transition models in determining the number of models ($N \leftarrow N^p$), then there will be no ambiguity and each model will

correspond to exactly one of the cases: the system stays in one subsystem, or the system switches from one subsystem to another.

As an example, we consider the following switched linear system:

$$\xi(t+1) = \begin{cases} \begin{bmatrix} -0.5 & 1 \\ -0.9 & 0 \end{bmatrix} \xi(t) + \begin{bmatrix} 0.8 \\ 0.3 \end{bmatrix} u(t) & \text{if subsystem 1} \\ \begin{bmatrix} -0.7 & 1 \\ -0.3 & 0 \end{bmatrix} \xi(t) + \begin{bmatrix} 0.2 \\ 0.4 \end{bmatrix} u(t) & \text{if subsystem 2} \end{cases} \tag{5.71}$$

$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \xi(t) \tag{5.72}$$

For this system, the input-output models are (can be verified using the procedure shown in Section 5.4):

$$y(t+1) = -0.5y(t) - 0.9y(t-1) + 0.8u(t) + 0.3u(t-1) \tag{5.73}$$

if the system stays in subsystem 1;

$$y(t+1) = -0.7y(t) - 0.3y(t-1) + 0.2u(t) + 0.4u(t-1) \tag{5.74}$$

if the system stays in subsystem 2;

$$y(t+1) = -0.7y(t) - 0.9y(t-1) + 0.2u(t) + 0.3u(t-1) \tag{5.75}$$

when switching from subsystem 1 to subsystem 2; and

$$y(t+1) = -0.5y(t) - 0.3y(t-1) + 0.8u(t) + 0.4u(t-1) \tag{5.76}$$

when switching from subsystem 2 to subsystem 1.

The initial state $\xi(0)$ is set as zero. We generated 100 data points $y(1) \ldots y(100)$. The control input $u(0) \ldots u(99)$ is generated randomly in the range of $[0, 10]$. The subsystem which is active at a particular instant is also chosen randomly with equal probability.

The simulation result by using the "Least Geometrical Mean Algorithm" from Chapter 2 is as follows: If we had used only two identification models for the two subsystems, not aware of the existence of the transition models, we were not able to correctly identify the system parameters. However, if we use four models as described above to cover also the two additional transition models, we were able to obtain perfect identification, as shown in Table 5.1.

From the above example, it can be readily concluded that for a good identification of switched system, we must fix enough (i.e. N^p) models to cover not only the usual input-output models (i.e. when the system stays in one subsystem for a long time), but also the transition models.

5.5.3 Identification of Switched Nonlinear Systems using Multiple Models

For the identification of switched nonlinear system, we can also use the multiple model approach, as shown in Fig. 5.5, with the identification models replaced by multilayer-perceptrons (MLP).

Consider the following switched nonlinear system:

$$\xi_1(t+1) = \begin{cases} \dfrac{5(1-\exp(-\xi_2(t)))}{1+\exp(-\xi_2(t))} & \text{if subsystem 1} \\ 0.1\xi_1(t) + \dfrac{2(u(t)+\xi_2(t))}{1+(u(t)+\xi_2(t))^2} & \text{if subsystem 2} \end{cases}$$

$$\xi_2(t+1) = \begin{cases} 0.1\xi_1(t)\xi_2(t) + \dfrac{(6+\xi_1(t))u(t)}{6} & \text{if subsystem 1} \\ 0.1\xi_2(t) + \dfrac{u(t)(2+u^2(t))}{1+\xi_1^2(t)+\xi_2^2(t)} & \text{if subsystem 2} \end{cases}$$

$$y(t) = \xi_1(t) + \xi_2(t) \tag{5.77}$$

It can be shown that (5.77) satisfies assumption 5.2. Therefore, there exists $N^p =$

Table 5.1: Parameter Estimates of a Switched System with Two Subsystems

Iterations	a11	a12	b11	b12
0 (randomly generated)	0.3993	0.7857	0.5059	0.5624
1	-0.7742	-0.6712	0.4785	0.4501
2	-0.7860	-0.9261	0.6568	0.3799
...	...	...	...	...
6	-0.5000	-0.8986	0.8002	0.2992
7	-0.5000	-0.9000	0.8000	0.3000
8	-0.5000	-0.9000	0.8000	0.3000
Iterations	a21	a22	b21	b22
0 (randomly generated)	0.0138	0.9875	0.1740	0.4030
1	-0.2858	-0.3686	0.1383	0.4020
2	-0.3690	-0.2429	0.1321	0.5347
...	...	...	...	...
6	-0.6995	-0.2996	0.2000	0.3998
7	-0.7000	-0.3000	0.2000	0.4000
8	-0.7000	-0.3000	0.2000	0.4000
Iterations	a31	a32	b31	b32
0 (randomly generated)	0.0694	0.1874	0.3490	0.9143
1	-1.3195	-0.8629	0.7445	0.4713
2	-1.0171	-0.8996	0.4526	0.3935
...	...	...	...	...
6	-0.7102	-0.8991	0.2033	0.3074
7	-0.7000	-0.9000	0.2000	0.3000
8	-0.7000	-0.9000	0.2000	0.3000
Iterations	a41	a42	b41	b42
0 (randomly generated)	0.7012	0.2854	0.9866	0.2243
1	-0.7264	-0.3493	0.7884	0.3534
2	-0.6594	-0.3409	0.8410	0.3372
...	...	...	...	...
6	-0.5001	-0.3000	0.8000	0.4000
7	-0.5000	-0.3000	0.8000	0.4000
8	-0.5000	-0.3000	0.8000	0.4000

$2^2 = 4$ input-output models in the neighborhood of the origin as follows:

$$y(t+2) = F_{11}(y(t), y(t+1), u(t), u(t+1)) \tag{5.78}$$

if the system stays in subsystem 1;

$$y(t+2) = F_{22}(y(t), y(t+1), u(t), u(t+1)) \tag{5.79}$$

if the system stays in subsystem 2;

$$y(t+2) = F_{21}(y(t), y(t+1), u(t), u(t+1)) \tag{5.80}$$

if $\sigma(t) = 1$ and $\sigma(t+1) = 2$; and

$$y(t+2) = F_{12}(y(t), y(t+1), u(t), u(t+1)) \tag{5.81}$$

if $\sigma(t) = 2$ and $\sigma(t+1) = 1$.

Starting from $\xi_1(0) = 0$ and $\xi_2(0) = 0$, we generated a training set with 2000 data points using (5.77), whereby the input $u(t)$ is a random number in the range of $[0, 1]$. The subsystem which is active at a particular instant is also chosen randomly with equal probability.

We make use of four MLP's to approximate the four functions in (5.78)-(5.81). Each MLP has four input nodes with $y(t+1)$, $y(t)$, $u(t+1)$ and $u(t)$ as inputs, one hidden layer with nine nodes, and one output node with $\hat{y}(t+2)$ as the output. The activation function for the hidden layer is the sigmoidal function, whereas the activation function for the output layer is identity. The MLP training algorithm we use is the "weighted back propagation" approach which has been detailed in chapter 2.6.1.

The simulation result is as follows: We observed from our simulation that if we were unaware of the existence of the transition models (5.80) and (5.81), and had used only two MLP's for the identification of the switched system (5.77), the test result is rather

poor. However, if we use four MLP's as described above, the identification performance improves significantly. After the MLP's have been trained for 10000 epochs with a learning rate $\eta_0 = 0.1$, we generated for each subsystem in (5.77) 40 test data (without switching). Two of the four MLP's indeed accurately approximate the nonlinear functions (Fig. 5.6).

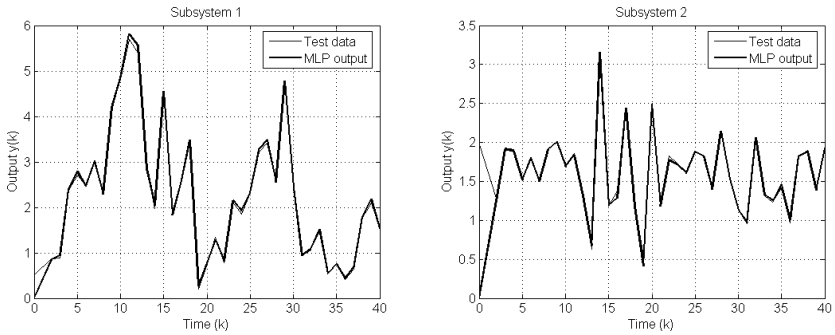

Figure 5.6: Simulation results for the identification of switched nonlinear system using multiple perceptrons

We thus conclude that transition models do exist for switched nonlinear systems, and that we shall use enough (i.e. N^p) MLP's for accurate identification of the switched system.

5.6 Conclusions

The existence of the transition models is an important phenomenon which must not be overlooked if we want to describe a switched system perfectly using input-output models. In this chapter, we derived mathematically the transition models for switched linear and nonlinear systems to describe the system behavior during the switching instants. The advantage of the proposed procedure is that it can be applied for general multiple-input-multiple-output systems, and for large classes of nonlinear systems. We also showed that these transition models are invariant under coordinate transformation of the state

space model. All these developments are established as rigorous results. The importance of these transition models in the design of controllers and system identification is also verified through simulation studies.

Chapter 6

Conclusions

6.1 Main Contributions

When nonlinear systems are required to work over a large operating range, a single linear controller will give very poor control performance or will even lead to instability of the whole system. One simple way to achieve a good control of the nonlinear system over its whole operating range, which at the same time retains the advantages of linear control design such as simplicity, is to separate the complete operating range into a few regions, design linear controllers for each of these regions, and then finally activate the corresponding controller when the system operates in a particular partition. This intuitive idea comes in various guises such as gain scheduling, supervisory control, and multiple model control, and has found success in process industry as well as flight control.

This dissertation focuses on the design of multiple model controller for nonlinear systems, as introduced above. To this aim, we first proposed to approximate the nonlinear system using piecewise affine ARX model, because this model structure automatically fits into the framework of multiple model control. The development of the model identification algorithm is detailed in chapters 2 and 3. In chapter 2, we presented the multiple model architecture for the identification of discrete-time time-varying systems.

The time-variation is approximated by a piecewise-constant function assuming a finite number N of unknown values, and thus the change in the system parameters is equivalent to switching from one subsystem to another. The subsystems of these switching systems can be either linear in the parameters or nonlinear. For the linear case, the multiple models are the usual linear regression model, whereas for the nonlinear case, multiple multilayer perceptrons are used to perform function approximation. We showed that by minimizing appropriate cost functions, i.e. the geometrical mean or the harmonic mean of the squares of errors, we can identify the subsytems with sufficient accuracy. We were able to derive both offline (batch mode) and online (sequential mode) methods for the system identification. Extensive simulation results verified the efficacy as well as robustness of our proposed algorithm. We also obtained a preliminary convergence proof of the parameter estimates for the case with linear subsystems.

In chapter 3, we formally moved on to identify piecewise affine ARX models. The piecewise affine ARX model is a special case of the switching system, whereby the subsystems are affine function of the input-output data, and the switching is orchestrated by the position of the regression vector within the complete regressor space. We proposed using data classifiers such as multilayer perceptrons or support vector machines to estimate the partition of the regressor space which defines the switching rule. To avoid a causality issue in the later stage of controller design, we also proposed a "modified" regression vector whereby the most current control signal is omitted from the regressor space. Our extensive simulation studies indicated that our algorithm can indeed provide accurate identification of piecewise affine systems as well as good approximation for nonlinear systems. The efficacy of our algorithm was also demonstrated by true experimental studies on DC motor and single-link robotic arm.

The multiple model controller is then designed and the details are given in chapter 4. More specifically, the control signal is computed in a similar way as weighted one-step-ahead predictive control, by relying on the piecewise affine model. This can be easily done because the model is affine in control, and thus the control signal is evaluated easily as a function of the reference signal as well as measurement data. There is however an important issue here: It is essential that the correct subsystem be chosen while computing the control signal, because the control signal is calculated based on the model of the "assumed" active subsystem in such a way that the next step's predicted output equals the reference signal. If a wrong subsystem is assumed, it is clear that the tracking performance will be poor. Therefore, we proposed to make use of the estimated partition of the regressor space to aid in determining the active subsystem. Yet there appears another problem: The regression vector which is used to determine the active subsystem contains the yet-to-be-calculated control signal! In view of this, we proposed five methods to overcome this problem, and showed via simulations and experiments that some of these methods work out very well.

Chapter 5 deals with the issue of the number of input-output subsystems needed to describe a switching system whose underlying subsystems are state space models. As is well-known to control engineers, for each state space model, an input-output model can be derived easily using z-transform. However, it is less well-known and less studied that when switching from one state space model to another, additional input-output models are needed to describe the dynamics. We rigorously developed a rather simple, yet effective, procedure to derive the input-output models from switching state space models. We called the additional input-output models during switching as the "transition models", and we further proved that the models are invariant to coordinate transformations of

the states. The advantage of our approach is its relative simplicity (and thus an easily adoptable methodology), and its ready applicability for the typically more difficult classes of switched nonlinear systems and MIMO systems. The importance of including these transition models, or in other words defining the correct number of input-output subsystems, in the design of controllers and system identification was verified through simulation studies.

6.2 Suggestions for Future Work

Past research activities have laid a foundation for the future work. Based on the prior research, the following questions deserve further consideration and investigations.

1. In chapter 2, we proved that by using our proposed algorithm, the geometrical mean of error squares will decrease monotonically. However, a complete proof that it will decrease to the global minimum (zero) is lacking at this moment. Thus, one important future work will be to analyze the convergence property of our least geometrical mean algorithm in a rigorous manner. The same applies to the least harmonic mean algorithm.

2. There are nonlinear systems which cannot be satisfactorily represented by a concatenation of linear systems, such as those which cannot undergo Taylor expansion. It would be interesting to extend or modify our approach to accommodate such nonlinearities.

3. Even though the multiple model control scheme in chapter 4 showed good tracking performance in simulations and experimental studies, we have yet to mathematically verify the stability of the closed loop system. The stability proof for switching system is a very complicated issue itself even for those systems without control input. Thus, the level of complexity for the case with control is of magnitude higher. Nevertheless, with

more and more tools of analysis appearing in the literature, we strongly believe that we can obtain a stability proof of the controlled system in the future.

4. The stability proof of model predictive control, the one-step-ahead controller being a special case, relies on an accurate model of the system. In our work, the nonlinear system is approximated using piecewise affine model, and furthermore, the partition of the regressor space is estimated using multilayer perceptrons or support vector machines. This means that there will be some unavoidable approximation errors of the PWARX model. It is crucial to understand how much the approximation error will affect the performance and stability of our multiple model control scheme.

5. Method IV of solving the chicken-and-egg problem, detailed in chapter 4, offers good control performance because the control signal is computed by making sure that a certain subsystem is activated. However, the related optimization problem is a nonlinear program and requires a long computational time. As such, the method cannot be used for real-time systems if a high sampling frequency is needed. We shall therefore look for possible way of reducing the computational burden for solving this optimization problem.

6. We shall also dwell into some fundamental issues such as how best to determine the optimal number of affine subsystems in a PWARX model as well as the order of these subsystems to describe a nonlinear system.

In conclusion, the control of nonlinear systems using multiple model or piecewise affine models has emerged as a practically viable method. A systematic approach is needed to address the numerous mathematical questions encountered while using the method in practice. And this thesis represents a further step in that direction.

Appendix A

The Weighted Back Propagation

In this section we would like to show the derivation of the weighted back propagation algorithm.

A.1 The Multilayer Perceptron

Consider the m^{th} multilayer perceptron (MLP) with 1 input layer, 2 hidden layers and 1 output layer, as shown in Fig. A.1. For this network, there are $n0$ inputs, $n1$ nodes in the first hidden layer, $n2$ neurons in the second hidden layer, and one output node. The activation functions for the hidden layers are the sigmoidal function

$$\varphi^{(1)}(v) = \varphi^{(2)}(v) = \frac{1}{1+e^{-v}} \tag{A.1}$$

whereas the activation function for the output layer is the linear function

$$\varphi^{(3)}(v) = v \tag{A.2}$$

For this MLP, some of the signals are as follows:

$$v_j^{(1)} = \sum_{i=1}^{n0} w_{ji}^{(1)} x_i \tag{A.3}$$

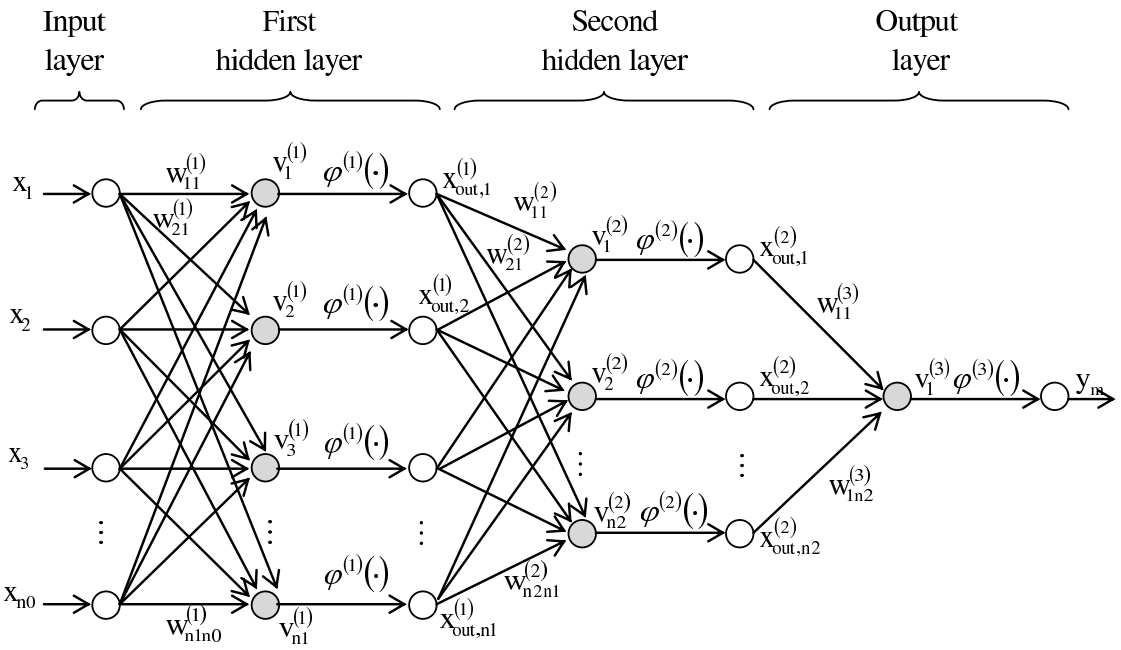

Figure A.1: A multilayer perceptron

$$x_{out,j}^{(1)} = \varphi^{(1)}(v_j^{(1)}) \tag{A.4}$$

$$v_j^{(2)} = \sum_{i=1}^{n1} w_{ji}^{(2)} x_{out,i}^{(1)} \tag{A.5}$$

$$x_{out,j}^{(2)} = \varphi^{(2)}(v_j^{(2)}) \tag{A.6}$$

$$v_1^{(3)} = \sum_{i=1}^{n2} w_{1i}^{(3)} x_{out,i}^{(2)} \tag{A.7}$$

$$y_m = v_1^{(3)} \tag{A.8}$$

A.2 Weight Updates

In the usual backpropagation algorithm (sequential mode), the following instantaneous error squares is to be minimized:

$$E = \frac{1}{2}(d - y)^2 = \frac{1}{2}e^2 \tag{A.9}$$

where d is the desired output.

To weigh the degree of updating the m^{th} MLP, we propose to minimize the following weighted error squares:

$$E = \frac{1}{2}p_m^2(d - y_m)^2 = \frac{1}{2}p_m e_m^2 \tag{A.10}$$

whereby we use p_m here as the weight for the error squares, instead of w as in earlier chapters, because w is used here to represent the weights within the MLP.

The weights of the MLP are then updated using the method of gradient descent

$$\Delta w_{ji}^{(s)} = -\eta^{(s)} \frac{\partial E}{\partial w_{ji}^{(s)}} \tag{A.11}$$

where s denotes the s^{th} layer of the MLP. The question now is how to obtain the partial derivative of E with respect to $w_{ji}^{(s)}$, and this can be considered layer by layer, as follows.

A.2.1 Weight Updates for Output Layer

For the output layer, we have

$$\frac{\partial E}{\partial w_{1i}^{(3)}} = \underbrace{\frac{\partial E}{\partial v_1^{(3)}}}_{I} \underbrace{\frac{\partial v_1^{(3)}}{\partial w_{1i}^{(3)}}}_{II} \tag{A.12}$$

$\partial E/\partial v_1^{(3)}$ is calculated as

$$\begin{aligned}\frac{\partial E}{\partial v_1^{(3)}} &= \frac{\partial}{v_1^{(3)}}\left(\frac{1}{2}p_m^2 e_m^2\right)\\ &= \frac{\partial}{v_1^{(3)}}\left(\frac{1}{2}p_m^2 (d-y_m)^2\right)\\ &= \frac{\partial}{v_1^{(3)}}\left(\frac{1}{2}p_m^2 (d-v_1^{(3)})^2\right)\\ &= -p_m^2\left(d-v_1^{(3)}\right)\\ &\triangleq -\delta_1^{(3)}\end{aligned} \tag{A.13}$$

whereas $\partial v_1^{(3)}/\partial w_{1i}^{(3)}$ is obtained as

$$\frac{\partial v_1^{(3)}}{\partial w_{1i}^{(3)}} = x_{out,i}^{(2)} \tag{A.14}$$

Therefore,

$$\frac{\partial E}{\partial w_{1i}^{(3)}} = -p_m^2\left(d-v_1^{(3)}\right)x_{out,i}^{(2)} = -\delta_1^{(3)}x_{out,i}^{(2)} \tag{A.15}$$

and

$$\Delta w_{1i}^{(3)} = \eta^{(3)}p_m^2\,(d-y_m)\,x_{out,i}^{(2)} = \eta^{(3)}\delta_1^{(3)}x_{out,i}^{(2)} \tag{A.16}$$

A.2.2 Weight Updates for the Second Hidden Layer

For the second hidden layer, we have

$$\frac{\partial E}{\partial w_{ji}^{(2)}} = \underbrace{\frac{\partial E}{\partial v_j^{(2)}}}_{I}\underbrace{\frac{\partial v_j^{(2)}}{\partial w_{ji}^{(2)}}}_{II} \tag{A.17}$$

The II part of the differentiation is straightforward:

$$\frac{\partial v_j^{(2)}}{\partial w_{ji}^{(2)}} = x_{out,i}^{(1)} \tag{A.18}$$

The I part of the differentation can be calculated as follows

$$\begin{aligned}
\frac{\partial E}{\partial v_j^{(2)}} &= \frac{\partial E}{\partial x_{out,j}^{(2)}} \frac{\partial x_{out,j}^{(2)}}{\partial v_j^{(2)}} \\
&= \frac{\partial}{\partial x_{out,j}^{(2)}} \left(\frac{1}{2} p_m^2 e_m^2 \right) \frac{\partial \varphi^{(2)}(v_j^{(2)})}{\partial v_j^{(2)}} \\
&= \frac{\partial}{\partial x_{out,j}^{(2)}} \left(\frac{1}{2} p_m^2 (d - y_m)^2 \right) \varphi^{(2)'}(v_j^{(2)}) \\
&= \frac{\partial}{\partial x_{out,j}^{(2)}} \left(\frac{1}{2} p_m^2 (d - v_1^{(3)})^2 \right) \varphi^{(2)'}(v_j^{(2)}) \\
&= \frac{\partial}{\partial x_{out,j}^{(2)}} \left(\frac{1}{2} p_m^2 (d - \sum_{j=1}^{n2} w_{1j}^{(3)} x_{out,j}^{(2)})^2 \right) \varphi^{(2)'}(v_j^{(2)}) \\
&= -\underbrace{p_m^2 (d - y_m)}_{\delta_1^{(3)}} w_{1j}^{(3)} \varphi^{(2)'}(v_j^{(2)}) \\
&= -\delta_1^{(3)} w_{1j}^{(3)} \varphi^{(2)'}(v_j^{(2)}) \\
&\triangleq -\delta_j^{(2)}
\end{aligned} \tag{A.19}$$

Therefore,

$$\frac{\partial E}{\partial w_{ji}^{(2)}} = -\delta_1^{(3)} w_{1j}^{(3)} \varphi^{(2)'}(v_j^{(2)}) x_{out,i}^{(1)} = -\delta_j^{(2)} x_{out,i}^{(1)} \tag{A.20}$$

and

$$\Delta_{ji}^{(2)} = \eta^{(2)} \delta_1^{(3)} w_{1j}^{(3)} \varphi^{(2)'}(v_j^{(2)}) x_{out,i}^{(1)} = \eta^{(2)} \delta_j^{(2)} x_{out,i}^{(1)} \tag{A.21}$$

A.2.3 Weight Updates for the First Hidden Layer

The derivation of the weight update rule for the first hidden layer is similar to that of the second layer. Again, we have

$$\frac{\partial E}{\partial w_{ji}^{(1)}} = \underbrace{\frac{\partial E}{\partial v_j^{(1)}}}_{I} \underbrace{\frac{\partial v_j^{(1)}}{\partial w_{ji}^{(1)}}}_{II} \tag{A.22}$$

with

$$\frac{\partial v_j^{(1)}}{\partial w_{ji}^{(1)}} = x_i \tag{A.23}$$

and

$$\begin{aligned}
\frac{\partial E}{\partial v_j^{(1)}} &= \frac{\partial E}{\partial x_{out,j}^{(1)}} \frac{\partial x_{out,j}^{(1)}}{\partial v_j^{(1)}} \\
&= \frac{\partial}{\partial x_{out,j}^{(1)}} \left(\frac{1}{2} p_m^2 e_m^2 \right) \frac{\partial \varphi^{(1)}(v_j^{(1)})}{\partial v_j^{(1)}} \\
&= \frac{\partial}{\partial x_{out,j}^{(1)}} \left(\frac{1}{2} p_m^2 (d - y_m)^2 \right) \varphi^{(1)'}(v_j^{(1)}) \\
&= \frac{\partial}{\partial x_{out,j}^{(1)}} \left(\frac{1}{2} p_m^2 (d - v_1^{(3)})^2 \right) \varphi^{(1)'}(v_j^{(1)}) \\
&= \frac{\partial}{\partial x_{out,j}^{(1)}} \left(\frac{1}{2} p_m^2 (d - \sum_{i=1}^{n2} w_{1i}^{(3)} x_{out,i}^{(2)})^2 \right) \varphi^{(1)'}(v_j^{(1)}) \\
&= \frac{\partial}{\partial x_{out,j}^{(1)}} \left(\frac{1}{2} p_m^2 \left(d - \sum_{i=1}^{n2} w_{1i}^{(3)} \varphi^{(2)} \left(v_i^{(2)} \right) \right)^2 \right) \varphi^{(1)'}(v_j^{(1)}) \\
&= \frac{\partial}{\partial x_{out,j}^{(1)}} \left(\frac{1}{2} p_m^2 \left(d - \sum_{i=1}^{n2} w_{1i}^{(3)} \varphi^{(2)} \left(\sum_{j=1}^{n1} w_{ij}^{(2)} x_{out,j}^{(1)} \right) \right)^2 \right) \varphi^{(1)'}(v_j^{(1)}) \\
&= - \underbrace{p_m^2 (d - y_m)}_{\delta_1^{(3)}} \left(\sum_{i=1}^{n2} w_{1i}^{(3)} \varphi^{(2)'}(v_i^{(2)}) w_{ij}^{(2)} \right) \varphi^{(1)'}(v_j^{(1)}) \\
&= - \left(\sum_{i=1}^{n2} \underbrace{\delta_1^{(3)} w_{1i}^{(3)} \varphi^{(2)'}(v_i^{(2)})}_{\delta_i^{(2)}} w_{ij}^{(2)} \right) \varphi^{(1)'}(v_j^{(1)}) \\
&= - \left(\sum_{i=1}^{n2} \delta_i^{(2)} w_{ij}^{(2)} \right) \varphi^{(1)'}(v_j^{(1)}) \\
&\stackrel{\triangle}{=} -\delta_j^{(1)}
\end{aligned} \tag{A.24}$$

Thus we have

$$\frac{\partial E}{\partial w_{ji}^{(1)}} = - \left(\sum_{i=1}^{n2} \delta_i^{(2)} w_{ij}^{(2)} \right) \varphi^{(1)'}(v_j^{(1)}) x_i = -\delta_j^{(1)} x_i \tag{A.25}$$

and

$$\Delta w_{ji}^{(1)} = \eta^{(1)} \left(\sum_{i=1}^{n2} \delta_i^{(2)} w_{ij}^{(2)} \right) \varphi^{(1)'}(v_j^{(1)}) x_i = \eta^{(1)} \delta_j^{(1)} x_i \tag{A.26}$$

A.2.4 Summary

As a summary, we have

$$\Delta w_{ji}^{(s)} = \eta^{(s)} \delta_j^{(s)} x_{out,i}^{(s-1)} \tag{A.27}$$

for all the layers. The difference lies in how the local gradient $\delta_j^{(s)}$ are computed. For the output layer, we have

$$\delta_1^{(output)} = p_m^2 (d - y_m) \tag{A.28}$$

whereas for the hidden layers, the local gradient is

$$\delta_j^{(s)} = \left(\sum_{i=1}^{n_{s+1}} \delta_i^{(s+1)} w_{ij}^{(s+1)} \right) \varphi^{(s)\prime}(v_j^{(s)}) \tag{A.29}$$

If we compare the above calculation of the local gradients with those of the conventional back propagation algorithm, we notice that the weight p_m on how much the MLP weights should be updated appears only in the local gradient of the output layer.

Appendix B

Published/Submitted Papers

Refereed Journal Articles:

[1] Xuemei Ren, Chow Yin Lai, Venkatakrishnan Venkataramanan, Frank L. Lewis, Suzhi Sam Ge, and Thomas Liew, "Feedforward Control Based on Neural Networks for Disturbance Rejection in Hard Disk Drives," *IET Control Theory and Applications*, Vol. 3, Issue 4, pp. 411-418, 2009.

[2] Chow Yin Lai, Frank L. Lewis, Venkatakrishnan Venkataramanan, Xuemei Ren, Suzhi Sam Ge, and Thomas Liew, "Disturbance and Friction Compensation in Hard Disk Drives using Neural Networks," *IEEE Trans. Industrial Electronics*, Vol. 57, No. 2, pp. 784-792, 2010.

[3] Chow Yin Lai, Cheng Xiang, and Tong Heng Lee, "Input-output Transition Models for Discrete-time Switched Linear and Nonlinear Systems," *Control and Intelligent Systems*, Vol. 39(1), pp. 47-59, 2011.

[4] Chow Yin Lai, Cheng Xiang, and Tong Heng Lee, "Identification and Control of Nonlinear Systems using Piecewise Affine Models," *IEEE Trans. Neural Networks, Special Issue on Data-Based Optimization, Modeling and Control*, submitted.

International Conference Articles:

[1] Chow Yin Lai, Frank L. Lewis, Venkatakrishnan Venkataramanan, Xuemei Ren, Suzhi Sam Ge, and Thomas Liew, "Neural Networks for Disturbance and Friction Compensation in Hard Disk Drives", in *Proc. IEEE Conference on Decision and Control*, Cancun, Mexico, 2008, pp. 3640-3645.

[2] Chow Yin Lai, Cheng Xiang, and Tong Heng Lee, "Input-output Transition Models for Discrete-time Switched Nonlinear Systems," in *Proc. IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, Singapore, July 2009, pp. 112-117.

[3] Cheng Xiang, Chow Yin Lai, Tong Heng Lee, and Kumpati S. Narendra , "A General Framework for Least-Squares Based Identification of Time-Varying System using Multiple Models," in *Proc. IEEE International Conference on Control and Automation 2009*, Christchurch, New Zealand, Dec. 2009, pp. 212-219.

[4] Chow Yin Lai, Cheng Xiang, and Tong Heng Lee, "Identification of Linear Time-Varying System using Multiple Models," in *Proc. IASTED International Conference on Modelling, Simulation and Identification (Robotics, Telematics and Applications)*, Beijing, China, Oct. 2009, paper 661-028.

[5] Chow Yin Lai, Cheng Xiang, and Tong Heng Lee, "Identification of Piecewise Affine Systems and Nonlinear Systems using Multiple Models," in *Proc. IEEE International Conference on Control and Automation 2010*, Xiamen, China, June 2010, pp. 2005-2012.

[6] Chow Yin Lai, Cheng Xiang, and Tong Heng Lee, "Identification and Control of Piecewise Affine Systems using Multiple Models and Model Predictive Control," in *Proc.*

IASTED International Conference on Control and Applications 2010, Banff, Canada, July 2010, pp. 530-537.

[7] Chow Yin Lai, Cheng Xiang, and Tong Heng Lee, "Identification and Control of Nonlinear Systems via Piecewise Affine Approximation," in *Proc. IEEE Conference on Decision and Control*, Atlanta, USA, December 2010, pp. 6395-6402.

[8] Chow Yin Lai, Cheng Xiang, and Tong Heng Lee, "Identification and Control of Nonlinear Systems using Piecewise Affine Models," in *Proc. International Conference on Control, Automation, Robotics and Vision*, Singapore, December 2010, pp. 2259-2265.

Bibliography

[1] A. Alessio and A. Bemporad, "Feasible mode enumeration and cost comparison for explicit quadratic model predictive control of hybrid systems," in *Proc. IFAC Conference on Analysis and Design of Hybrid Systems*, Alghero, Italy, 2006, pp. 302-308.

[2] B. Aufderheide, V. Prasad, and B. W. Bequette, "A comparison of fundamental model-based and multiple model predictive control," in *Proc. 40th IEEE Conference on Decision and Control*, Orlando, USA, Dec 2001, pp. 4863-4868.

[3] B. Aufderheide and B. W. Bequette, "Extension of dynamic matrix control to multiple models," *Computers and Chemical Engineering*, vol. 27, 2003, pp. 1079-1096.

[4] A. Bemporad, F. Borrelli, and M. Morari, "Piecewise linear optimal controllers for hybrid systems," in *Proc. American Control Conference*, Chicago, 2000, pp. 1190-1194.

[5] A. Bemporad, F. Borrelli, and M. Morari, "Optimal controllers for hybrid systems: Stability and piecewise linear explicit form," in *Proc. 39th IEEE Conference on Decision and Control*, Sydney, Australia, Dec 2000, pp. 1810-1815.

[6] A. Bemporad, A. Garulli, S. Paoletti, and A. Vicino, "A bounded-error approach to piecewise affine system identification," *IEEE Trans. Automatic Control*, vol. 50(10),

2005, pp. 1567-1580.

[7] A. Bemporad and M. Morari, "Control of systems integrating logic, dynamics, and constraints," *Automatica*, vol. 35, 1999, pp. 407-427.

[8] F. Borrelli, M. Baotic, A. Bemporad, and M. Moradi, "Dynamic programming for constrained optimal control of discrete-time linear hybrid systems," *Automatica*, vol. 41, 2005, pp. 1709-1721.

[9] E. F. Camacho, D. R. Ramirez, D. Limon, D. Munoz de la Pena, T. Alama, "Model predictive control techniques for hybrid systems," *Annual Reviews in Control*, vol. 34(1), pp. 21-31.

[10] Y. Cao and C. W. Silva, "Supervised switching control of a deployable manipulator system," *Control and Intelligent Systems*, vol. 34(2), 2006, pp. 153-165.

[11] M. Cavalletti, G. Ippoliti, and S. Longhi, "Lyapunov-based switching control using neural networks for a remotely operated vehicle," *International Journal of Control*, vol. 80(7), 2007, pp. 1077-1091.

[12] S. Chaib, D. Boutat, A. Benali, and F. Kratz, "Dynamic controller of switched linear systems: a common Lyapunov function approach," in *Proc. 45th IEEE Conference on Decision and Control*, San Diego, USA, Dec 2006, pp. 125-130.

[13] F. C. Chen and H. K. Khalil, "Adaptive control of a class of nonlinear discrete-time systems using neural networks," *IEEE Trans. Automatic Control*, vol. 40(5), 1995, pp. 791-801.

[14] Q. Chen, L. Gao, R. A. Dougal, and S. Quan, "Multiple model predictive control for a hybrid proton exchange membrane fuel cell system," *Journal of Power Sources*, vol. 191, 2009, pp. 473-482.

[15] J. Cho, J. C. Principe, D. Erdogmus, and M. A. Motter, "Quasi-sliding mode control strategy based on multiple-linear models," *Neurocomputing*, vol. 70(4-6), 2007, pp. 960-974.

[16] M. L. Corradini and G. Orlando, "An observer based switching control strategy for the stabilization of uncertain multi-input-multi-output plants" *International Journal of Control*, vol. 80(4), 2007, pp. 583-592.

[17] B. De Schutter and T. J. J. van den Boom, "MPC for continuous piecewise-affine systems,"' *Systems and Control Letters*, vol. 52, 2004, pp. 179-192.

[18] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society B*, vol. 39, 1977, pp. 1-37.

[19] M. J. Feiler and K. S. Narendra, "Simultaneous identification and control of time-varying systems," in *Proc. 45th IEEE Conference on Decision and Control*, San Diego, USA, Dec. 2006, pp. 1093-1098.

[20] G. Ferrari-Trecate, M. Muselli, D. Liberati, and M. Morari, "A clustering technique for the identification of piecewise affine systems," *Automatica*, vol. 39, 2003, pp. 205-217.

[21] Y. Fu, T. Y. Chai, and H. Yue, "Intelligent control using multiple models and neural networks," *International Journal of Adaptive Control and Signal Processing*, vol. 22(5), June 2008, pp.495-509.

[22] P. Grieder, M. Kvasnica, M. Baotic, and M. Morari, "Stabilizing low complexity feedback control of constrained piecewise affine systems," *Automatica*, vol. 41, 2005, pp. 1683-1694.

[23] B. Gu and Y. P. Gupta, "Control of nonlinear processes by using linear model predictive control algorithms," *ISA Transactions*, vol. 47, 2008, pp. 211-216.

[24] H. Haruno, D. W. Wolpert, and M. Kawato, "Mosaic model for sensorimotor learning and control," *Neural Computation*, vol. 13, 2003, pp. 2201-2220.

[25] A. Hassibi and S. Boyd, "Quadratic stabilization and control of piecewise-linear sytems," in *Proc. American Control Conference*, Philadelphia, USA, June 1998, pp. 3659-3664.

[26] W. Heemels, B. De Schutter, and A. Bemporad, "Equivalence of hybrid dynamical models," *Automatica*, vol. 37, 2001, pp. 1085-1091.

[27] J. Imura, "Optimal control of sampled-data piecewise affine systems," *Automatica*, vol. 40, 2004, pp. 661-669.

[28] A. Lj. Juloski, W. P. M. H. Heemels, and G. Ferrari-Trecate, "Data based hybrid modelling of the component placement process in pick-and-place machines," *Control Engineering Practice*, vol. 12, 2004, pp. 1241-1252.

[29] A. Lj. Juloski, S. Weiland, and W. P. M. H. Heemels, "A bayesian approach to identification of hybrid systems," *IEEE Trans. Automatic Control*, vol. 50(10), 2005, pp. 1520-1533.

[30] A. Juloski, W. P. M. H. Heemels, G. Ferrari-Trecate, R. Vidal, S. Paoletti, and J. H. G. Niessen, "Comparison of four procedures for the identification of hybrid systems," in *Hybrid systems: computation and control*, Vol. 3414 of Lecture Notes in Computer Science, M. Morari and L. Thiele, Eds. Springer Verlag, Berlin/Heidelberg, 2005, pp. 354-369.

[31] C. G. Kallstrom, K. J. Astrom, N. E. Thorell, J. Eriksson, and L. Stein, "Adaptive autopilots for tankers," *Automatica*, vol. 15, 1979, pp. 241-254.

[32] E. W. Kamen, P. P. Khargonekar, and K. R. Poolla, "A transfer-function approach to linear time-varying discrete-time systems," *SIAM Journal on Control and Optimization*, vol. 23(4), 1985, pp. 550-565.

[33] E. W. Kamen, Fundamentals of linear time-varying systems, in W. S. Levine (Ed.), *The Control Handbook*, (CRC Press and IEEE Press, 1996, 451-468).

[34] J. Kohlmorgen, S. Lemm, K. R. Müller, S. Liehr, and K. Pawelzik, "Fast change point detection in switching dynamics using a hidden Markov model of prediction experts," in *Proc. International Conference on Artificial Neural Networks*, 1999, pp. 204-209.

[35] K. Kong, H. C. Kniep, and M. Tomizuka, "Output saturation in electric motor systems: Identification and controller design," *ASME Journal of Dynamic Systems, Measurement, and Control*, accepted for publication.

[36] A. Kordon, P. S. Dhurjati, Y. O. Fuentes, and B. A. Ogunnaike, "An intelligent parallel control system structure for plants with multiple operating regimes," *Journal of Process Control*, vol. 9, 1999, pp. 453-460.

[37] D. J. Leith and W. E. Leithead, "Survey of gain-scheduling analysis and design," *International Journal of Control*, vol. 73(11), 2000, pp. 1001-1025.

[38] A. U. Levin and K. S. Narendra, "Control of nonlinear dynamical systems using neural networks - Part II: observability, identification and control," *IEEE Trans. Neural Networks*, vol. 7(1), 1999, pp. 30-42.

[39] D. Liberzon, J. P. Hespanha, and A. S. Morse, "Stability of switched linear systems: A Lie-algebraic condition," *Systems and Control Letter*, vol. 37(3), 1999, pp. 117-122.

[40] D. Liberzon and A. S. Morse, "Basic problems in stability and design of switched systems," *IEEE Control Systems Magazine*, vol. 19(5), 1999, pp. 59-70.

[41] D. Liberzon, *Switching in systems and control* (Birkhäuser, Boston, 2003).

[42] H. Lin and P. J. Antsaklis, "Stability and stabilizability of switched linear systems: a survey of recent results," in *Proc. IEEE International Symposium on Intelligent Control*, 2005, pp. 24-29.

[43] H. Lin and P. J. Antsaklis, "Switching stabilizability for continuous-time uncertain switched linear systems," *IEEE Trans. Automatic Control*, vol. 45(4), 2007, pp. 633-646.

[44] M. Margaliot, "Stability analysis of switched systems using variational principles: An introduction," *Automatica*, vol. 42(12), 2006, 2059-2077.

[45] M. Morari and M. Baric, "Recent developments in the control of constrained hybrid systems," *Computers and Chemical Engineering*, vol. 30, 2006, pp. 1619-1631.

[46] Maxon Motor AG (2008), *EC-powermax 30 Catalogue Infromation*, 188.

[47] A. S. Morse, "Supervisory control of families of linear set-point controllers - part 1: Exact matching," *IEEE Trans. Automatic Control*, vol. 41(10), Oct. 1996, pp. 1413-1431.

[48] A. S. Morse, "Supervisory control of families of linear set-point controllers - part 2: Robustness," *IEEE Trans. Automatic Control*, vol. 42(11), Nov. 1997, pp. 1500-1515.

[49] F. M'Sahli, C. Fayeche, and R. B. Abdennour, "Multiple model and multiple control approaches: Application to a semi-batch reactor," *Control and Intelligent Systems*, vol. 37(4), 2009, pp. 195-203.

[50] H. Nakada, K. Takaba, and T. Katayama, "Identification of piecewise affine systems based on statistical clustering technique," *Automatica*, vol. 41, 2005, pp. 905-913.

[51] N. N. Nandola and S. Bhartiya, "A multiple model approach for predictive control of nonlinear hybrid systems," *Journal of Process Control*, vol. 18, 2008, pp. 131-148.

[52] N. N. Nandola and S. Bhartiya, "Hybrid sysem identification using a structural approach and its model based control: An experimental validation," *Hybrid Systems*, vol. 3, 2009, pp. 87-100.

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

[54] K. S. Narendra and J. Balakrishnan, "Improving transient response of adaptive control systems using multiple models and switching," *IEEE Trans. Automatic Control*, vol. 39, Sep. 1994, pp. 1861-1866.

[55] K. S. Narendra and S. M. Li, "Neural networks in control systems," in *Mathematical perspectives on neural networks*, P. Smolensky, M. C. Mozer, and D. E. Rumelhart, Eds. Lawrence Erlbaum Associates, London.

[56] K. S. Narendra and J. Balakrishnan, "Adaptive control using multiple models and switching," *IEEE Trans. Automatic Control*, vol. 42, Feb. 1997, pp. 171-187.

[57] K. S. Narendra and C. Xiang, "Adaptive control of discrete-time systems using multiple models," *IEEE Trans. Automatic Control*, vol. 45, Sep. 2000, pp. 1656-1668.

[58] L. Ozkan, M. V. Kothare, and C. Georgakis, "Model predictive control of nonlinear systems using piecewise linear models," *Computers and Chemical Engineering*, vol. 24, 2000, pp. 793-799.

[59] L. Ozkan and M. V. Kothare, "Stability analysis of a multi-model predictive control algorithm with application to control of chemical reactors," *Journal of Process Control*, vol. 16, 2006, pp. 81-90.

[60] S. Paoletti, A. L. Juloski, G. F. Tracate, and R. Vidal, "Identification of hybrid systems: a tutorial," *European Journal of Control*, vol. 513(2-3), 2007, pp. 242-260.

[61] K. Pawelzik, J. Kohlmorgen, and K. R. Müller, "Annealed competition of experts for a segmentation and classification of switching dynamics," *Neural Computation*, vol. 8(2), 1996, pp. 340-356.

[62] B. Pregelj and S. Gerksic, "Hybrid explicit model predictive control of a nonlinear process approximated with a piecewise affine model," *Journal of Process Control*, vol. 20, 2010, pp. 832-839.

[63] A. Rantzer and M. Johansson, "Piecewise linear quadratic optimal control," *IEEE Trans. Automatic Control*, vol. 45(4), 2000, pp. 629-637.

[64] L. Rodrigues, A. Hassibi, and J. P. How, "Output feedback controller synthesis for piecewise-affine systems with multiple equilibria," in *Proc. American Control Conference*, Chicago, USA, June 2000, pp. 1784-1789.

[65] L. Rodrigues and J. P. How, "Automated control design for a piecewise-affine approximation of a class of nonlinear systems," in *Proc. American Control Conference*, Arlington, USA, June 2001, pp. 3189-3194.

[66] L. Rodrigues and J. P. How, "Synthesis of piecewise-affine controllers for stabilization of nonlinear systems," in *Proc. 42nd IEEE Conference on Decision and Control*, Maui, USA, Dec 2003, pp. 2071-2076.

[67] J. Roll, A. Bemporad, and L. Ljung, "Identification of piecewise affine systems via mixed-integer programming," *Automatica*, vol. 40, 2004, pp. 37-50.

[68] J. Roll, A. Nazin, and L. Ljung, "Nonlinear system identification via direct weight optimization," *Automatica*, vol. 41, 2005, pp. 475-490.

[69] W. J. Rugh and J. S. Shamma, "Research on gain scheduling," *Automatica*, vol. 36, 2000, pp. 1401-1425.

[70] A. V. Savkin and R. J. Evans, *Hybrid dynamical systems: Controller and sensor switching problems* (Birkhäuser, Boston, 2002).

[71] R. N. Shorten and K. S. Narendra, "Necessary and sufficient conditions for the existence of a CQLF for a finite number of stable LTI systems," *International Journal of Adaptive Control and Signal Processing*, vol. 16(10), 2002, pp. 709-728.

[72] E. D. Sontag, "Nonlinear regulation: the piecewise linear approach," *IEEE Trans. Automatic Control*, vol. 26(2), 1981, pp. 346-358.

[73] G. Stein, G. L. Hartmann, and R. C. Hendrick, "Adaptive control laws for F-8 flight test," *IEEE Trans. Automatic Control*, vol. 22(5), 1977, pp. 758-767.

[74] Z. Sun and S. S. Ge, *Switched linear systems: Control and design* (Springer-Verlag, 2005).

[75] Z. Sun, "Stabilizing switching design for switched linear systems: A state-feedback path-wise switching approach, *Automatica*, vol. 45(7), 2009, 1708-1714.

[76] J. Thomas, S. Olaru, J. Buisson, and D. Dumur, "Robust model predictive control for piecewise affine systems subject to bounded disturbances", *Analysis and Design of Hybrid Systems*, 2006, pp. 329-334.

[77] R. Vidal, S. Soatto, Y. Ma, and S. Sastry, "An algebraic geometric approach to the identification of a class of linear hybrid systems," in *Proc. 42nd IEEE Conference on Decision and Control*, Maui, Hawaii USA, Dec. 2003, pp. 167-172.

[78] R. Vidal, "Recursive identification of switched ARX systems," *Automatica*, vol. 44(9), 2008, pp. 2274-2287.

[79] C. Wen, S. Wang, X. Jin, X. Ma, "Identification of dynamic systems using Piecewise-Affine basis function models," *Automatica*, vol. 43, 2007, pp. 1824-1831.

[80] K. Wulff, F. Wirth, and R. Shorten, "A control design method for a class of switched linear systems," *Automatica*, vol. 45, 2009, pp. 2592-2596.

[81] C. Xiang, "Existence of global input-output model for nonlinear systems," in *Proc. 2005 International Conference on Control and Automation*, Budapest, Hungary, 2005, pp. 125-130.

[82] Z. R. Xiang and W. M. Xiang, "Observer design for a class of switched nonlinear systems," *Control and Intelligent Systems*, vol. 36(4), 2008, pp.318-322.

[83] P. Yan and H. Ozbay, "On switching H-infinity controllers for a class of linear parameter varying systems," *Systems and Control Letters*, vol. 56, 2007, pp. 504-511.

[84] T. W. Yoon, J. S. Kim, and A. S. Morse, "Supervisory control using a new control-relevant switching" *Automatica*, vol. 43(10), Oct. 2007, pp. 1791-1798.

[85] W. Zhang and L. Yu, "Stability analysis for discrete-time switched time-delay system," *Automatica*, vol. 45(10), 2009, pp. 2265-2271.

[86] W. Zhang, A. Abate, J. Hu, and M. P. Vitus, "Exponential stabilization of discrete-time switched linear system," *Automatica*, vol. 45(11), 2009, pp. 2526-2536.

[87] T. Zou, X. Wang, S. Y. Li, and Q. M. Zhu, "A mixed logic enhanced multi-model switching predictive controller for nonlinear dynamical process," *Control and Intelligent Systems*, vol. 35(2), 2007, pp. 154-161.