Waseda University Doctoral Dissertation

Study on Identification of Nonlinear Systems Using Quasi-ARX Models

Yu CHENG

Graduate School of Information, Production and Systems
Waseda University

June 2012

To my parents and my friends,
for all those times you stood by me;
for all the truth that you made me see;
for all the joy you brought to my life;
for all the wrong that you made right;
for every dream you made come true;
for all the love I found in you.

I'll be forever thankful for you.

Abstract

System identification can be used to construct a model to represent a given system, and it plays an important role in system analysis, control and prediction. From the view of application, conventional nonlinear black-box models are not good since an easy-to-use model is to interpret properties of the nonlinear process, rather than treated as vehicles for adjusting the fit to the data. Therefore, some careful modeling is needed for certain applications, and prior knowledge from system is inspired to be combined with formal properties of the model.

Quasi-linear autoregressive with exogenous inputs (quasi-ARX) modeling scheme provides an effective approach to extend well-studied and user-friendly linear techniques to nonlinear applications. It constructs models consisting of two parts: a macro-part and a core-part. The macro-part owns a useful interface to introduce some properties favorable to specific applications, and the core-part is a flexible nonlinear model to parameterize complicated coefficients of the macro-part. To this end, an ARX-like linear structure is constructed as the macro-part by using Taylor expansion; while the coefficients are parameterized by a multi-input-multi-output (MIMO) nonlinear model in the core-part.

Nevertheless, it is no easy solution to identify nonlinear systems using the quasi-ARX models, though it is equipped with a useful structure. Followed by requirements of real applications, the identification is expected to interpret properties of the nonlinear system and hold the principle of simplicity. One effective approach to this challenge is to divide the model parameters into two parts: the nonlinear parameters and the linear parameters. The nonlinear parameters mean those interpretable ones, such as translation and dilation parameters of wavelet basis function, which can be determined by using prior knowledge. When the nonlinear parameters are fixed, the quasi-ARX model can be transformed linear in parameters. These linear parameters are the ones to fit the data, which can be estimated by linear regression methods. Furthermore, the quasi-ARX model is meaningful to nonlinear polynomial system identification, which often contains a big number of candidate monomial terms. The identified quasi-ARX model inspires a pre-processing approach to evaluate significance of each monomial term, which is helpful to reduce the candidate pool efficiently. In this thesis, investigations are firstly made on nonlinear parameter estimation with clustering partition and grid partition method, where wavelet network (WN) and neurofuzzy network (NFN) are included as the core-part of the model, respectively. Then the linear parameter is estimated by means of kernel method, where radial basis function network (RBFN) is incorporated. Finally, neural network (NN) is embedded in the quasi-ARX model, which is identified and provides a pre-screening scheme for polynomial system identification.

Identification of the quasi-ARX model using clustering partition is proposed by incorporating WN into the model core-part. An automatic clustering algorithm is used to obtain prior knowledge of data distribution, where input space is partitioned into reasonable number of clusters, and translation and dilation parameters of each wavelet basis function are estimated in terms of heuristic guides in the associated subspace. In this way, the interpretable parameters of the quasi-ARX model can be determined and fixed by means of prior knowledge efficiently. It is meaningful not only from the aspect of nonlinear system identification but also in nonlinear adaptive control, where the quasi-ARX WN predictor is utilized and identified by the proposed method. Reference signal can be tracked quickly since only linear parameters are needed to be adjusted, even a sudden change is happened on the system.

Compared with clustering partition method, identification of the quasi-ARX model using grid partition provides a simple approach to obtain prior knowledge. Typically, the partition made on each input dimension can be explained in the way of fuzzy membership partition, and the generated fuzzy rules work as the prior knowledge to unveil the system properties. To this end, NFN is incorporated into the quasi-ARX model, where a set of linguistic fuzzy rules are represented by neurofuzzy basis functions in the NFN. However, it suffers from curse-of-dimensionality, which may result in over-fitting from redundant fuzzy rules. Heuristically, the quasi-ARX model input variables which are linear with the output, are sufficient to describe the input-output relationship by linear expression, thus are less important to parameterize coefficients of the ARX-like linear structure. Based on this fact, fuzzy rules are reduced by using only necessary inputs for the incorporated NFN, where linear correlation between each model input and output is considered, and it is combined in fitness function of genetic algorithm (GA) in the form of a modified Bayesian information criterion (BIC).

With prior knowledge obtained from the above two approaches, the quasi-ARX model can be identified using kernel learning method, where the model is transformed to a support vector regression (SVR) with a composite kernel. The SVR based identification introduces robust performance for linear parameter estimation. Moreover, it also provides an efficient approach to cope with curse-of-dimensionality when grid partition is utilized. Instead of estimating the big number of linear parameters directly, a dual form of quadratic programming (QP) optimization is implemented, where the complexity of representation by support vectors is independent of input dimension. At the same time, explicit and physically meaningful kernel mapping is proposed, which is learnt by means of the quasi-ARX modeling with prior knowledge. It leads to an appropriate high-dimensional feature space, where nonlinearity of mapping can be adjusted according to complexity of the model core-part. In this way, the associated kernel function is called quasi-linear kernel, which is categorized into composite kernels with the nonlinearity between linear and some existed nonlinear kernel functions.

Identification of the quasi-ARX model can also be used for nonlinear polynomial systems, which are often composed of a huge monomial candidate pool, while a parsimonious model structure is finally expected to represent the system dynamics. In most cases, it is a difficult problem and time-consuming for optimization based methods. Consider the fact that the quasi-ARX NN model has provided impressive fitting performance and can prevent the training from local minima, it is

applied to provide an index for pre-screening selection, which may improve identification efficiency by reducing candidate term pool to an reasonable size. In the pre-screening step, a quasi-ARX NN model is identified to approximate the system under study initially; then a Taylor expansion of the identified model is performed, and the importance of each monomial term is evaluated according to variance of the term with its coefficient. In the following step, multi-objective evolutionary algorithm (MOEA) is used to determine the appropriate model structure in the reduced searching space efficiently. Both the model approximation ability and complexity are optimized simultaneously.

Preface

The common theme of this thesis is studying on identification of nonlinear systems by means of the quasi-ARX models. The material is organized in six chapters. Most of the material has been published or considered to publish in journal papers and conference papers.

The material in Chapter 2 can be found in

- Yu Cheng, Lan Wang and Jinglu Hu, "Quasi-ARX Wavelet Networks for Nonlinear System Identification", in *Proc. of 2010 International Conference on Modeling, Simulation and Control (ICMSC'10)*, pp. 407–411, Cairo, Egypt, Nov., 2010.

which has been extended into a journal paper

- Yu Cheng, Lan Wang and Jinglu Hu, "Quasi-ARX Wavelet Network for SVR Based Nonlinear System Identification", *Nonlinear Theory and its Applications (NOLTA), IEICE, Vol. 2, No. 2*, pp. 165–179, 2011.

The material in Chapter 3 can be found in

- Yu Cheng, Lan Wang, Jing Zeng and Jinglu Hu, "Identification of Quasi-ARX Neurofuzzy Model by Using SVR-based Approach with Input Selection", in *Proc. of 2011 IEEE International Conference on Systems, Man and Cybernetics (SMC'11)*, pp. 1585–1590, Anchorage, U.S., Oct., 2011.

which has been extended into a journal paper

- Yu Cheng, Lan Wang and Jinglu Hu, "Identification of Quasi-ARX Neurofuzzy Model with an SVR and GA Approach", *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E95-A, No. 5*, pp. 876–883, 2012.

The material in Chapter 4 can be found in

- Yu Cheng and Jinglu Hu, "Nonlinear System Identification Based on SVR with Quasi-linear Kernel", in *Proc. of 2012 International Joint Conference on Neural Networks (IJCNN)*, Brisbane, Australia, June, 2012, (to appear).
- Yu Cheng, Lan Wang and Jinglu Hu, "A Quasi-linear Approach for Microaary Missing Value Imputation", in *Proc. of 18th International Conference on Neural Information Processing (ICONIP'2011)*, pp. 233–240, Shanghai, China, Nov., 2011.

- Yu Cheng, Lan Wang and Jinglu Hu, "Quasi-ARX Wavelet Network for SVR Based Nonlinear System Identification", *Nonlinear Theory and its Applications (NOLTA), IEICE, Vol. 2, No. 2*, pp. 165–179, 2011.

The material in Chapter 5 can be found in

- Yu Cheng, Miao Yu, Lan Wang and Jinglu Hu, "An Efficient Identification Scheme for Nonlinear Polynomial NARX Model", in *Proc. of 16th International Symposium on Artificial Life and Robotics (AROB 16th'11)*, pp. 499–502, Beppu, Japan, Jan., 2011.
- Yu Cheng, Lan Wang and Jinglu Hu, "A Two-step Method for Nonlinear Polynomial Model Identification Based on Evolutionary Optimization", in *Proc. of World Congress on Nature and Biologically Inspired Computing (NaBIC 2009)*, pp. 613–618, Coimbatore, India, Dec., 2009.

which have been extended into a journal paper

- Yu Cheng, Lan Wang and Jinglu Hu, "A Two-step Scheme for Polynomial NARX Model Identification Based on MOEA with Pre-screening Process", *IEEJ Transactions on Electrical and Electronic Engineering (TEEE), Vol. 6, No. 3*, pp. 253–259, 2011.

Acknowledgements

First and foremost I would like to express my deep and sincere gratitude to my supervisor, Professor Jinglu Hu. His wide knowledge and his logical way of thinking have been of great value for me. I appreciate all his contributions of time, ideas, and funding to make my Ph.D experience productive and stimulating.

I am also very grateful to Professor Osamu Yoshie and Professor Shigeru Fujimura who proof-read all the thesis. Many thanks are for their valuable comments.

I wish to express my warm and sincere thanks to Professor Jianping Wang, Ph.D, Associate Dean of School of Electric Engineering and Automation, Hefei University of Technology, Hefei, China, who introduced me to the field of machine learning, and Professor Haiwen Liu, Ph.D, vice president of East China Jiao Tong University, Nanchang, China, who gave me important guidance during my first steps into Ph.D studies and gave me untiring help during my difficult moments.

I am indebted to Dr. Boyang Li, Dr. Benhui Chen and Dr. Lan Wang for their constructive and fruitful discussions in the start of my research.

I warmly thank a great group of friends at Waseda University. This includes my classmates in Neurocomputing Lab at Waseda University (Yang Chen, Wenxiang Dou, Bo Zhou, Zhan Shi, Yuling Lin, Imam Sutrisno et al.), and many other students and faculty, too numerous to name.

Finally, I owe my loving thanks to my parents: Bin Cheng and Ke Qiu. Without your encouragement and love, it would have been impossible for me to finish this work. Thank you.

Kitakyushu, Japan — Yu CHENG
February 11, 2012

Table of Contents

List of Tables

List of Figures

Glossary

Some notations may have different meaning locally.

Notations

x^T	transpose
$y(t), y^*(t), \hat{y}(t)$	output signal, reference output and prediction output at time t
$\bar{y}$	mean value
$u(t)$	input signal at time t
$e(t)$	noise at time t
$\varphi(t)$	regression vector
$g(\varphi(t))$	an unknown nonlinear function of $\varphi(t)$
$\Phi(t)$	regression vector containing nonlinear regressors
$\theta(\varphi(t))$	coefficient vector which is function of input-output variables
Θ	parameter vector
$\widehat{\Theta}$	estimation of Θ
$\boldsymbol{p}_j$	parameter vector of the 'basis functions' in nonlinear nonparametric model
Ω_{ij}	coordinate parameters of nonlinear nonparametric model
n_y	number of old output values in $\varphi(t)$
n_u	number of old input values in $\varphi(t)$
n	number of old input and output values in $\varphi(t)$, $n = n_y + n_u$
M	number of 'basis functions' in nonlinear nonparametric model
ξ_t, ξ_t^*	slack variables of SVR

Operators and Functions

$\|\cdot\|$	norm
$\|\cdot\|$	absolute
$<\cdot,\cdot>$	inner production
q^{-1}	the backward shift operator, $q^{-1}f(t) = f(t-1)$
$\otimes$	Kronecker production
$\prod$	production
$\sum$	summation

$\mu_{A_i^j}$	fuzzy membership function of fuzzy set A_i^j
$\mathcal{N}$	'basis function' of the nonlinear nonparametric model
$\mathcal{K}$	kernel function
iff	if and only if
$\prec$	dominate

Abbreviations

AIC	Akaike's Information Criterion
AP	Affinity Propagation
ARX	AutoRegressive model structure with eXogenous input
BIC	Bayesian Information Criterion
EA	Evolutionary Algorithm
ERR	Error Reduction Ratio
GA	Genetic Algorithm
Inf	Infinite
InNFN	Incorporated NeuroFuzzy Network
LS	Least Square
LLSimputation	Local Least Square Imputation
MOEA	Multi-objective Evolutionary Algorithm
MSE	Mean Square Error
NaN	Not a number
NARX	Nonlinear AutoRegressive model structure with eXogenous input
NFN	NeuroFuzzy Network
NN	Neural Network
NNM	Nonlinear Nonparametric Model
NRMSE	Normalized Root Mean Square Error
NSGA-II	Non-dominated Sorting Genetic Algorithm-II
OLS	Orthogonal Least Square
Q-ARX-NFN	Quasi-ARX NeuroFuzzy Network
Q-ARX-NN	Quasi-ARX Neural Network
Q-ARX-WN	Quasi-ARX Wavelet Network
QP	Quadratic Programming
RBFN	Radial Basis Function Network
RMSE	Root Mean Spare Error
SISO	Single-Input and Single-Output
SVM	Support Vector Machine
SVR	Support Vector Regression
VC Dimension	Vapnik-Chervomenkis Dimension
WN	Wavelet Network

Chapter 1

Introduction and Motivation

1.1 System Identification

In loose terms a system is an object in which variables of different kinds interact and produce observable signals. Constructing models from observed data is a fundamental element in science, and system identification deals with the problem of building mathematical models of dynamical systems based on observed data from the system [1, 2]. The main feature of dynamical systems is that the future depends on the past. Thus a prediction of the output can be represented by all or some previous measured inputs and outputs.

1.1.1 Model Structures

The large family of models for system identification can be classified along several different aspects, and in many cases the dividing line is not sharp, but it is anyway a useful approach to get a grip on the various possibilities [3, 4, 5].

Linear vs. Nonlinear Models: Linear theory has been well developed and devoted to real applications during the last 30 years. Under the assumption of utilizing the linear structure for easy-use, a linear model uses parameters that are constant and do not vary throughout a simulation, such as autoregressive with exogenous inputs (ARX) model. This means that we can estimate one fixed value for the parameter at the beginning of the simulation and it will remain the same throughout. In contrast, a non-linear model introduces dependent parameters that are allowed to vary throughout the course of a simulation run, and its use becomes necessary where interdependencies between parameters cannot be considered insignificant.

Parametric vs. Nonparametric Models: Postulate a parameterized model set, which is parameterized by coefficients θ, and then adjust θ to minimize the least square fit between the model and

system. These models are named parametric models, including polynomial models and rational models. Non-parametric models, such as neural networks (NNs) [6, 7, 8], neurofuzzy networks (NFNs) [9, 10, 11], radial basis function networks (RBFNs) [12, 13, 14], wavelet networks (WNs) [15, 16, 17], are restricted to some parameterized function class, however this class must be sufficiently flexible, and the number of parameters is not a priori fixed. In some cases, the distinction between these two kinds of models may be difficult.

Global vs. Local Models: A local model is typically formed by local basis functions with bounded support which vanish rapidly at points far from the centers, such as radial basis functions and wavelets. A global model basically uses all the data for each estimation such as polynomial models and sigmoid neural networks. The similarity among these two types of models is that many of them own linear-in-parameter model structures, which are linear combinations of model terms or basis functions. It has been proven that linear-in-the-parameter models possess broad approximation capabilities and they have been widely used in time-series prediction, nonlinear system modeling and identification, signal processing and pattern recognition [18, 19].

White-box vs. Black-box Models: A white-box model is the case when a model is perfectly known, and it has been possible to construct it entirely from prior knowledge and physical insight. However, a black-box model is estimated from data without using any specific insight into how the data were generated, but the chosen model structure belongs to families that are known to have good flexibility and have been 'successful in the past'. In this thesis we concentrate on the term black-box models, and details will be given in the following subsection.

1.1.2 Model Selection and Parameter Estimation

From a set of candidate model structures collection, we are going to look for a suitable one. This is no doubt the most important and, at the same time, the most difficult choice of system identification. In black-box models, no physical insight is available or used, and a common assumption is that the system is linear. This is an attractive idea because the linear framework is simple and well studied. Furthermore, linear models are easy to be interpreted and understood, which require significantly less effort than the estimation of nonlinear models. Nevertheless, the assumption is never true, and there has been a tendency towards nonlinear black-box modeling in various application fields during the last decades. Neural network (NN) models, support vector regression (SVR) [20, 21, 22, 23], polynomial nonlinear autoregressive with exogenous inputs (NARX) models [18, 7] et al. have been proposed as state-of-art nonlinear black-box models for their outstanding approximation ability.

According to the model selected, the parameter estimation can be categorized into optimization

methods and constructive methods [7]. The optimization methods estimate the parameter vector using nonlinear optimization techniques. Specifically, the local search algorithms such as back-propagation for sigmoid NNs and structure risk minimization for SVR are termed to local optimization methods [24, 25]; while the global search algorithms such as genetic algorithm (GA) for polynomial NARX model [26, 27, 28, 29] are termed to global optimization methods. In contrast, the constructive methods are usually based on selection of appropriate basis functions from a finite set, and it is considered efficient to linear-in-parameter models. A typical case of such method is orthogonal least square (OLS) algorithm for polynomial NARX model parameter estimation [18, 30, 31].

However, the problem of the above nonlinear estimation methods is the unknown parameters are basically viewed as vehicles for adjusting the fit to the data thus rarely reflect physical considerations in the systems. From the view of application it is not good since an easy-to-use model is to interpret the properties of the process that it represents and to extract the knowledge of the underlying system [32]. It inspired prior knowledge has to be combined with formal properties of the model, and some careful modeling is needed for a model structure favorable to certain applications [1].

1.2 Quasi-ARX Model for System Identification

1.2.1 Quasi-ARX Modeling

Consider the fact that system identification is always followed by certain applications, conventional nonlinear black-box models have been criticized for not user-friendly since they neglect some good properties of popular linear black-box modeling, such as the linear structure and simplicity [33, 34]. Especially, the linear structure is useful and favorable to certain applications such as nonlinear system control and fault diagnosis [35, 36, 37, 38, 39]. Take nonlinear adaptive control as an example, the linear structure of the ARX model provides a convenient approach for controller design, where the linearity for input variables makes sharing parameters between prediction model and controller possible [2]. In contrast, the state-of-art nonlinear control methods, such as NNs based control approaches, meet the problem of complex controller design which has to contain two parts: the one used for predictor and the other used for controller [40, 41]. It also makes the adaptive control difficult to be realized.

To obtain nonlinear models oriented to applications, it is natural to consider a modeling scheme and construct models consisting of two parts: a macro-part and a core-part [33, 34]. The macro-part is a user-friendly interface constructed to introduce some properties favorable to specific applications, while embedding the resulted model complexity in the coefficients. The core-part is a flexible

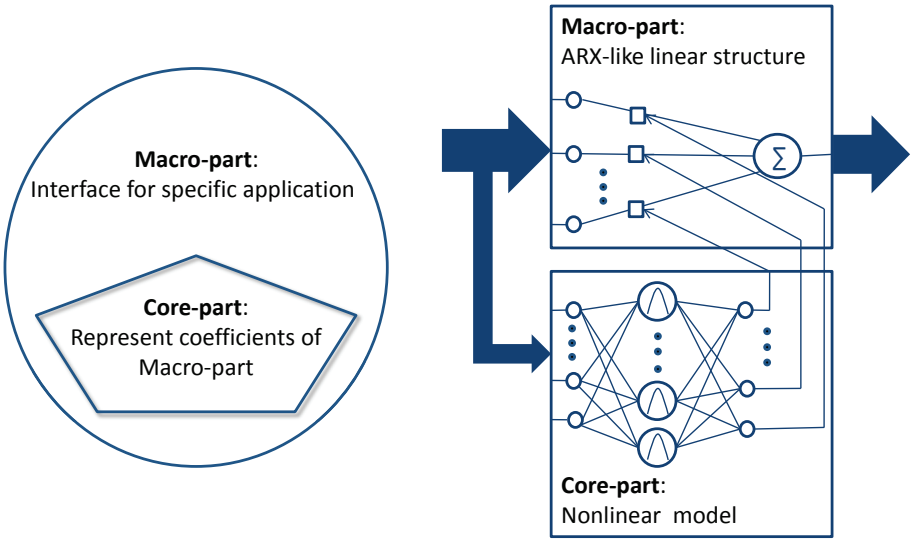


Figure 1.1: Quasi-ARX modeling: Basic idea of the quasi-ARX modeling is shown in the left figure, where a macro-part and a core-part are included in the constructed model. An example is illustrated in the right figure. An ARX-like linear structure is worked as the macro-part for specific application, whose coefficients are parameterized by a flexible MIMO nonlinear model.

nonlinear model, which is used to represent the complicated coefficients of the macro-part. The model constructed in this way can be shown in Fig.1.1, and it is expected to be user-friendly and has impressive approximation ability. To this end, by using Taylor expansion or other mathematic transformation techniques, a class of ARX-like interfaces are constructed as macro-parts, in which useful linear properties are introduced, while the complexities are embedded in the coefficients. Then multi-input-multi-output (MIMO) nonlinear models such as NNMs are used as the core-part to parameterize these coefficients. Thanks for the resultant quasi-ARX model, the quasi-ARX predictor can be designed linear with input variable for the aforementioned nonlinear adaptive control, where an ARX-like macro-part is worked as a linear structure, and the input variable $u(t)$ in the coefficients can be skillfully replaced by an extra variable. In other words, the quasi-ARX modeling provides an alternative approach to extend well-studied linear techniques to nonlinear applications.

1.2.2 Prior Knowledge

Followed by application-oriented modeling basic idea, the quasi-ARX model is expected to interpret the properties of the nonlinear system and hold the principle of simplicity for identification.

Therefore, some parameters in the core-part should be determined and fixed by using knowledge information, which is, in fact, interpretable in the basis functions, and always available or can be obtained via some ways in practice. On the other hand, from the general view of system identification, determining a model from a finite sample of observations without any prior knowledge about the system is an ill-posed problem, in the sense that a unique model may not exist, or it may not depend continuously on the observations [42]. It is therefore highly motivated to incorporate prior knowledge into the quasi-ARX models to determine some parameters interpretable in the core-part. For this purpose, two approaches for the prior knowledge are introduced.

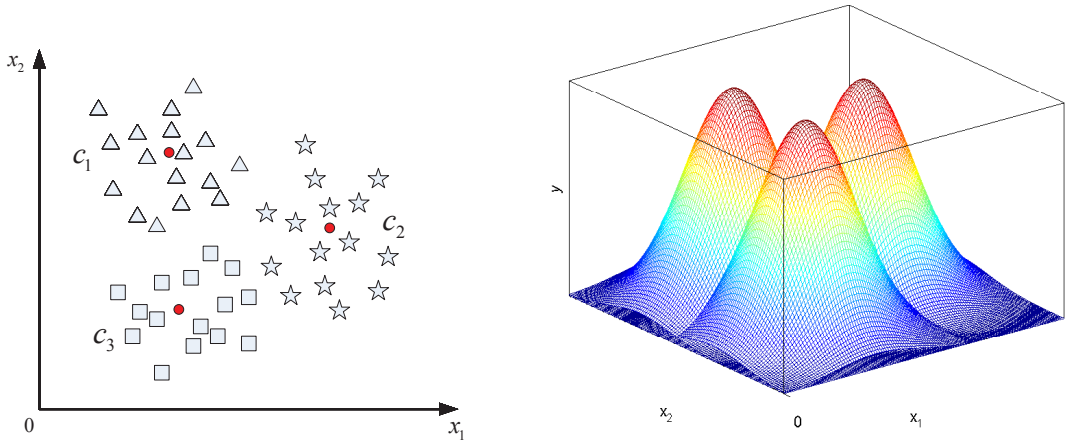
Clustering Partition Data distribution is one of the most important information from a nonlinear system. It is utilized as prior knowledge by clustering methods, where the input space is partitioned and the center of each cluster is set as position parameters of basis function models, such as RBFNs and WNs. For example, in a two-dimensional case, three clusters are formed and the centers are denoted by C_1, C_2 and C_3 as shown in Fig.1.2(a). Then a two-dimensional RBFN model can be constructed as in Fig.1.2(b), which owns the fixed center parameters of C_1, C_2, C_3.

Grid Partition Grid partition is a simple way to obtain system properties. The partition is made on each input dimension, which can be explained in the way like fuzzy membership partition, and the generated fuzzy rules as the prior knowledge can unveil the system dynamics to some extend. An example of two-dimensional case using RBF fuzzy membership function is illustrated in Fig.1.3, where two inputs x_1 and x_2 are divided into four uniform parts for simplicity, and the input space is covered by fuzzy membership functions with center parameters of $\overline{x} = [\overline{x}_1^1, \ldots, \overline{x}_1^4, \overline{x}_2^1, \ldots, \overline{x}_2^4]$. With these fixed parameters , the total number of fuzzy rules is $M = 4 \times 4$.

It is known that gird partition method works in a simple way, however, it is easy to be suffered from curse-of-dimensionality problem. In contrast, although clustering partition method is insensitive to the input dimension, the accurate clustering is not an easy job, and the parameter determination is lack of rational guide in the quasi-ARX models.

1.2.3 Parameter Estimation Methods

Parameter estimation of the quasi-ARX model is difficult since the the estimation methods should be performed favorable to certain applications efficiently. It corresponds to the concept of ”learning” in the machine learning area.



(a) Samples in the input space (b) Interpolation for each local area

Figure 1.2: An example of clustering partition

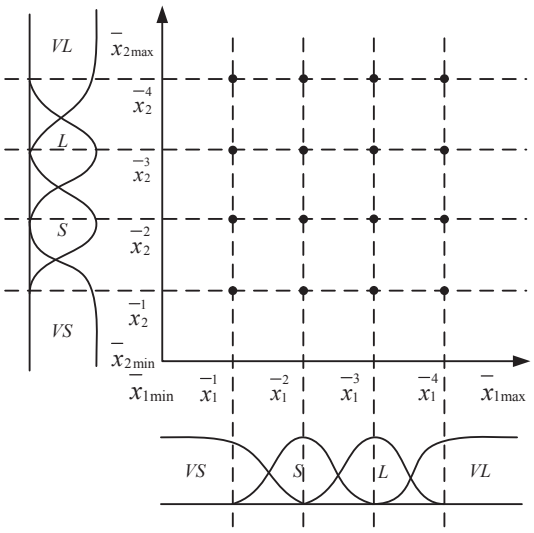


Figure 1.3: An example of grid partition

In the situation that nonlinear basis function models such as WNs and NFNs are utilized to parameterize coefficients of the macro-part, the quasi-ARX model can be explained in the manner of multi-local linear with interpolation [43, 44, 45], where the core-part is considered in the sense of interpolation for the corresponding local linear model. In this way, some parameters in the model are interpretable, such as dilation and translation parameters of WNs, or partition parameter of operating region in NFNs, which can be determined using prior knowledge provided above;

such pre-determined parameters result in a multi-local linear model with fixed interpolations, which endow the quasi-ARX model linear-in-parameter property. Simple linear estimation algorithms like least square (LS) algorithm can be utilized for the estimation efficiently.

When global models such as NNs are incorporated in the quasi-ARX model [46, 47], the model can be expressed as $y(t) = \varphi^T(t)\theta + \varphi^T(t)W_2\Gamma(W_1\varphi(t) + B) + e(t)$, where $y(t)$ and $e(t)$ are the system output and disturbance, $\varphi(t)$ is the regressor, θ and W_1, W_2, B are parameters of the model. Γ is the diagonal nonlinear operator with identical sigmoidal elements. Thus the model is interpreted in two parts: linear part and nonlinear part. The linear part is shown as the first term and treated as an ARX model when W_1, W_2 and B are fixed; the nonlinear part is the second term, which is a NN interpolated multimodel under the condition that θ is fixed. A hierarchical estimation algorithm is introduced, and parameters in this two parts are estimated alternatively. It is worthy of note that the dual-loop estimation may help BP algorithm for NN training to get out of local minima.

What's more, support vector regression (SVR) has been used for parameter estimation when NNMs are incorporated in the quasi-ARX model [48]. Without any prior knowledge, the nonlinear functions of the NNMs are computed in an implicit way by means of kernel technique, while the remaining weight parameters can be learnt based on Vapnik's ε-insensitive loss function and structural risk minimization [25, 49, 22, 21]. A key point in this method is the use of kernel techniques, which perform nonlinear mapping to a high-dimensional feature space implicitly by replacing the inner product with a positive definite kernel function [23, 50, 51]. However, from the view of applications, such kernel trick with the implicit kernel mapping is not always good because:

1. It is difficult to choose appropriate kernel from a limited number of existed kernel functions for some certain applications. Although the composite kernel functions satisfy the Mercer's condition and lead to many choices of hybrid kernels, it is not easy work which depends on the wit of the users and their understanding of the processing data.

2. The implicit nonlinear kernel mapping may face potential over-fitting problems for some complex and noised learning task. For instance, in functional genomics and microarray data some tasks are nonlinear in nature, with characteristics of high noise, and large number of input features compared with the relatively small number of training examples. It is found that the SVMs with implicit nonlinear kernel functions such as Gaussian kernel are severely over-fitting at times, thus can not obtain good results and even perform less than the linear model [52, 53].

1.3 Challenges and Obstacles

The quasi-ARX models have been proposed from the perspective of real applications, and applied successfully in nonlinear adaptive control and fault detection. However, system identification using such models still faces following challenges and obstacles:

- *Clustering partition based prior knowledge acquisition*
  Clustering partition based prior knowledge including data distribution information is useful in the quasi-ARX model for system identification. It is obviously that accurate clustering partition method and rational guide for determination of the interpretable parameters are helpful to improve the model performance. What's more, the number of basis functions in NNMs plays an important role in model construction. Insufficient basis functions can not capture the nonlinear dynamics, but redundant ones may deteriorate the generalization capability. Though these issues are important for the identification, limited effort has been put in them.

- *Curse-of-dimensionality*
  Grid partition by using prior knowledge is easy to suffer from curse-of-dimensionality. For instance, the grid partition by incorporating NFNs into the quasi-ARX models, which corresponds to fuzzy membership partition, is carried out on each input dimension. Fuzzy rules are generated by combining fuzzy membership functions of all the inputs, hence are increased exponentially with input dimension. It may result in high computational complexity and over-fitting.

- *Kernel learning for system identification*
  SVR based identification method is famous for generalization and robust performance, where main task lies in selection of kernel functions. However, most of the existed kernels take advantage of the 'kernel trick' to reproduce feature space, which work in an implicit way and suffer from potential over-fitting. A physically meaningful and explicit kernel mapping is desired and challenged using the quasi-ARX model with interpretable parameters and prior knowledge. Hopefully, it is expected to encourage a more efficient and extensive application of kernel learning methods in nonlinear system identification.

- *High complexity in nonlinear polynomial system identification*
  Identification of nonlinear polynomial systems face the problem of huge size of monomial candidate terms, which increases exponentially with number of input variables and model

nonlinearity. It makes successful evolutionary computation based structure selection algorithms work at low efficiency. Consider the fact that the quasi-ARX NN model has provided impressive fitting performance and can prevent training from local minima , it is challenged to provide an index for pre-screening selection, which may improve the identification efficiency by reducing the candidate term pool to an reasonable size.

1.4 Thesis Outlines and Main Contributions

This thesis consists of six chapters. Chapter 1 gives background and outline for the whole thesis. Chapter 2 and Chapter 3 identify the quasi-ARX models using clustering partition and grid partition method, respectively. In Chapter 4, the quasi-ARX model is identified by means of kernel learning approach. Chapter 5 utilizes the quasi-ARX NN model for nonlinear polynomial system identification. Finally, Chapter 6 gives a summary for the whole thesis. The flow of this thesis is depicted in Fig.1.4.

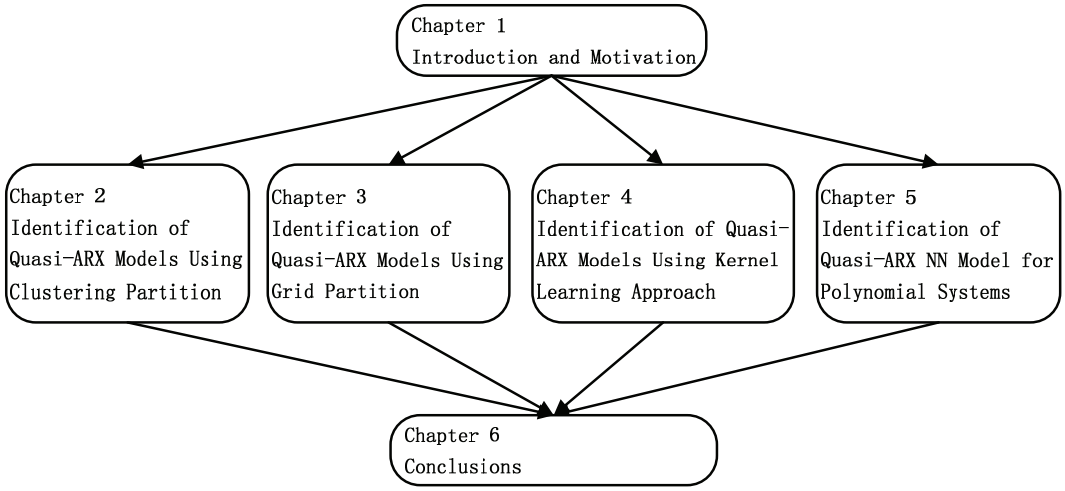


Figure 1.4: Flow diagram of this thesis

Chapter 2: Chapter 2 identifies the quasi-ARX model by introducing clustering partition based prior knowledge, where WN is incorporated into the model core-part. Prior knowledge is utilized to guide dilation and translation parameters in the WN heuristically, which is obtained from an automatical clustering algorithm, and number of basis functions in the WN becomes deterministic at the same time. The main contributions of this chapter are concluded as following:

- An heuristic guide is proposed for interpretable parameters determination in the quasi-ARX WN model, which is based on prior knowledge from an automatic clustering algorithm.
- Dilation and translation parameters of the quasi-ARX WN model can be fixed during adaptive control process, thus only linear parameters are needed to be adjusted to cope with sudden changes on the system.

Chapter 3: Chapter 3 is devoted to solving curse-of-dimensionality problem when NFN is incorporated into the quasi-ARX models. It is a typical problem of the quasi-ARX model identification using grid partition method. The number of fuzzy rules increases exponentially with the input space dimension, which may result in an over-complex model. Although some input-dimension-insensitive methods may alleviate computational burden to identify such a complex model, redundant fuzzy rules are still remained, and over-fitting is easy to be caused.

Heuristically, the model input variables which are linear with the output, are sufficient to describe the input-output relationship by linear expression, thus are less important to appear in the core-part of the quasi-ARX models. Based on this fact, the linear correlation between each model input and output is calculated and considered as a factor in fitness function to select significant inputs of the incorporated NFN, where the model complexity is trade-off against fitting performance by a modified Bayesian information criterion (BIC). As a global and easy selection method, GA is utilized in this chapter. In this way, the model generalization ability is improved.

Chapter 4: Chapter 4 is devoted to identifying the quasi-ARX model by using kernel learning approach, in which the quasi-ARX model is transformed to an SVR with composite kernel. The contributions of SVR based identification method can be concluded as:

- SVR based identification introduces robust performance for parameter estimation of the quasi-ARX model since the utilization of structure risk minimization.
- SVR provides an efficient approach to cope with curse-of-dimensionality when grid partition method is utilized to generate prior knowledge as Chapter 3 shown. The linear parameter estimation of the quasi-ARX model is transformed to a dual form of quadratic programming (QP) optimization in the reproduced feature space, where the complexity of representation by support vectors is independent of input dimension.

Moreover, explicit and physically meaningful kernel mapping is proposed, which depends on the quasi-ARX modeling with prior knowledge. The associated kernel function is called quasi-linear kernel, which can be categorized into a composite kernel with a novel learning method. It is distinctive to others in the following issues:

- The quasi-linear kernel is learnt by using prior knowledge of certain applications, which leads to an interpretable kernel mapping and appropriate high-dimensional feature space.
- The nonlinearity (complexity) of the kernel mapping is tunable between the existed linear and nonlinear kernel methods, thus can cope with over-fitting from the complex and noised systems.

Chapter 5: Chapter 5 proposes a quasi-ARX NN model based pre-screening method for nonlinear polynomial system identification. Nonlinear polynomial system identification often faces the problem of huge size of candidate pool, which makes the optimization based methods work in low efficiency. To solve this problem, a two-step identification scheme is proposed, where the selection based pre-screening step is implemented firstly to form a relative small candidate term pool with all the necessary monomial terms included. Then optimization based identification is implemented efficiently. The contributions of this chapter are shown as following:

- Two Importance Index are proposed to select important monomial terms. The first one is estimated by using a quasi-ARX NN model to approximate the system under study initially; then a Taylor expansion of the identified model is performed, and the importance of each monomial term is evaluated according to variance of the term with its coefficient. The other is calculated by error reduction ratio (ERR) from a simplified OLS algorithm, which is applied to further select terms with big contribution to the system output.
- Multi-objective evolutionary algorithm (MOEA) is used to determine the appropriate model structure in the reduced searching space efficiently. Both the model approximation ability and complexity are optimized simultaneously.

Chapter 2

Identification of Quasi-ARX Models Using Clustering Partition

2.1 Introduction

Quasi-ARX model possesses an ARX-like linear structure, where flexible nonlinear nonparametric model (NNM) is applied to interpret the complicated coefficients of the linear structure. It makes the model easy to be used in nonlinear adaptive control and fault detection [36, 35]. However, incorporation of NNs still suffers from the lack of efficient constructive methods, both for determining the parameters of neurons and for choosing network structure [16]. What's more, from the view of nonlinear adaptive control, it is desired that sudden changes on the system can be tracked by adjustment of only a part of parameters of the model easily. Nevertheless, if NNs are embedded into the quasi-ARX model, no parameter could be fixed beforehand since they are lack of meaningful explanations, thus needed to be trained repeatedly for any change of the system.

On the other hand, wavelet network (WN) is a kind of special feedforward neural networks supported by wavelet theory, where wavelets are introduced as active functions of hidden neurons with a linear output neuron [15, 54, 55]. Recently, nonlinear system identification by WNs is attracting a growing interest [1, 56, 17], not only due to the universal approximation ability of WNs, but also for the efficient constructive methods, both for determining parameters and for choosing network structure [16, 57, 58]. Therefore, it seems natural to incorporate WN into the quasi-ARX model, where interpretable parameters can be determined by prior knowledge from the system.

In this chapter, dilation and translation, which are nonlinear to the quasi-ARX wavelet network (Q-ARX-WN) model, are determined and fixed by means of clustering partition method. Clustering algorithm is performed to capture data distribution information, which is used to guide the parameter

determination heuristically [57, 58]. In this way, the Q-ARX-WN model is constructed to show linear property in three aspects. Firstly, the Q-ARX-WN model is equipped with a linear ARX-like structure, where the nonlinear part is embedded into the coefficients to improve the overall flexibility and compromise with simplicity. The representation of this part is in Section 2.3.1. Secondly, the Q-ARX-WN predictor could be linear in input variable $u(t)$ which makes a controller can be easily obtained from the well-known linear control theory, and details are also discussed in Section 2.3.3. Thirdly, the Q-ARX-WN predictor is linear-in-parameter when the nonlinear parameters are determined heuristically. It is useful in nonlinear adaptive control since the interpretable parameters of the Q-ARX-WN predictor can be fixed even subtle change has happened on the system, thereby the control is continued with only a part of parameters adjusted in a linear way.

This chapter is organized as follows: Section 2.2 formulates the problem. In Section 2.3, the Q-ARX-WN model and predictor are introduced in detail. In Section 2.4, the clustering partition method is proposed for the Q-ARX-WN model parameter estimation. Simulation studies are carried out in Section 2.5. Section 2.6 presents some discussions, and the conclusions are summarized at last.

2.2 Problem Description

Consider a single-input-single-output (SISO) nonlinear time-invariant system whose input-output dynamics is described as

$$\begin{aligned} y(t) &= g(\varphi(t))+e(t) \\ \varphi(t) &= [y(t-1),\cdots,y(t-n_y),u(t-d),\cdots,u(t-n_u-d+1)]^T \end{aligned} \tag{2.2.1}$$

where $u(t) \in R$, $y(t) \in R$, $e(t) \in R$ are the system input, the system output and a stochastic noise of zero-mean at time t ($t = 1, 2, \cdots$), respectively. $g(\cdot) : R^{n=n_u+n_y} \rightarrow R$ is an unknown function (black-box) describing the dynamics of system under study, and $\varphi(t) \in R^n$ is the regression vector composed of delayed input-output data. d is the known integer time delay, n_u and n_y are unknown maximum delays of the input and output respectively, and n is the number of input variables, which equals to the sum of n_u and n_y.

2.3 Quasi-ARX Prediction Model

In this part, the quasi-ARX prediction model is introduced, which possesses an ARX-like linear structure, and useful in generating a predictor linear in the input variable $u(t)$.

2.3.1 Regression Form of the Quasi-ARX Model

Assumption 1 : $g(\cdot)$ is a continuous function, but at $\varphi(t) = 0$ it is C^∞ continuous.

Performing Taylor expansion to the unknown nonlinear function $g(\varphi(t))$ around the region $\varphi(t) = 0$

$$y(t) = g(0) + g'(0)\varphi(t) + \frac{1}{2}\varphi^T(t)g''(0)\varphi(t) + \cdots . \tag{2.3.1}$$

Since $g(\cdot)$ is assumed to be continuously differentiable, the derivative $g^{(i)}(0)(i = 1, 2, \cdots)$ exists. Then ignoring $g(0)$ for simplicity and a regression form could be generated as

$$y(t) = \varphi^T(t)\theta(\varphi(t)) + e(t) \tag{2.3.2}$$

where

$$\begin{aligned} \theta(\varphi(t)) &= (g'(0) + \frac{1}{2}\varphi^T(t)g''(0) + \cdots) \\ &= [a_{1,t}, \cdots, a_{n_y,t}, b_{0,t}, \cdots, b_{n_u-1,t}]^T. \end{aligned} \tag{2.3.3}$$

Here, the coefficients $a_{i,t} = a_i(\varphi(t))$ and $b_{i,t} = b_i(\varphi(t))$ are nonlinear functions of $\varphi(t)$.

2.3.2 Quasi-ARX Predictor

In order to predict $y(t)$ by input-output data up to time $t - d$ in a prediction model, we need that the coefficients of polynomials are calculable using the input-output data up to time $t - d$. To do so, let us replace iterative $y(t - i), i = 1, 2, \cdots, d - 1$ with their predictions, and new expression of the coefficients $a_{i,t}$ and $b_{i,t}$ in Eq.(2.3.3) are

$$a_{i,t} \triangleq \widetilde{a}_i(\phi(t - d)), b_{i,t} \triangleq \widetilde{b}_i(\phi(t - d)) \tag{2.3.4}$$

where $\phi(t - d) = q^{-d}\phi(t)$, and q^{-1} is the backward shift operator, e.g. $q^{-1}u(t) = u(t - 1)$. $\phi(t)$ is defined by

$$\phi(t) = [y(t), \cdots, y(t - n_y + 1), u(t), \cdots, u(t - n_u - d + 2)]^T. \tag{2.3.5}$$

Next, two polynomials are introduced based on the coefficients, which are defined as

$$A(q^{-1}, \phi(t)) = 1 - a_{1,t}q^{-1} - \cdots - a_{n_y,t}q^{-n_y} \tag{2.3.6}$$

$$B(q^{-1}, \phi(t)) = b_{0,t} + b_{1,t}q^{-1} + \cdots + b_{n_u-1,t}q^{-n_u+1}. \tag{2.3.7}$$

then expression (2.2.1) could be represented as

$$A(q^{-1},\phi(t-d))y(t) = y_0 + B(q^{-1},\phi(t-d))q^{-d}u(t) + e(t). \tag{2.3.8}$$

Finally, it could be proved that the d-step-ahead prediction satisfies [34]

$$y^o(t+d|t,\phi(t)) = y_\phi + \alpha(q^{-1},\phi(t))y(t) + \beta(q^{-1},\phi(t))u(t) \tag{2.3.9}$$

where

$$\begin{aligned}
&y^o(t+d|t,\phi(t)) = y(t+d) - F(q^{-1},\phi(t))e(t+d),\\
&y_\phi = F(q^{-1},\phi(t))y_0,\\
&\alpha(q^{-1},\phi(t)) = G(q^{-1},\phi(t)) = \alpha_{0,t} + \alpha_{1,t}q^{-1} + \cdots + \alpha_{n_y-1,t}q^{-(n_y-1)},\\
&\beta(q^{-1},\phi(t)) = F(q^{-1},\phi(t))B(q^{-1},\phi(t)) = \beta_{0,t} + \beta_{1,t}q^{-1} + \cdots + \beta_{n_u+d-2,t}q^{-n_u-d+2}
\end{aligned}$$

and $G(q^{-1},\phi(t)), F(q^{-1},\phi(t))$ are unique polynomials satisfying

$$F(q^{-1},\phi(t))A(q^{-1},\phi(t)) = 1 - G(q^{-1},\phi(t))q^{-d}$$

2.3.3 Predictor Linear in Input Variable $u(t)$

It is found in Eq.(2.3.9) that the prediction model is nonlinear in $u(t)$, and from the view of controller design, it is not good to use. To solve this problem, another assumption is given.

Assumption 2 : The system is controllable, and the controller could be expressed by $u(t) = \rho(\xi(t))$, where $\xi(t) = [y(t),\cdots,y(t-n_y),u(t-1),\cdots,u(t-n_u),y^*(t+d)]^T$, and $y^*(t)$ denotes desired output.

To do so, $y^*(t)$ is introduced to replace the variable $u(t)$ in $\phi(t)$, thus the coefficients $y_\phi, \alpha_{i,t}, \beta_{j,t}$ ($i = 0,\cdots,n_y-1,\ j = 0,\cdots,n_u+d-2$) are functions of $\xi(t)$, and the predictor is linear in $u(t)$. From [34], this predictor is useful for controller design and described by

$$y^o(t+d|t,\xi(t)) = y_\xi + \alpha(q^{-1},\xi(t))y(t) + \beta(q^{-1},\xi(t))u(t) \tag{2.3.10}$$

where y_ξ is a coefficient which is unknown function of $\xi(t)$.

Therefore, the quasi-ARX predictor could be expressed as

$$\begin{aligned}
y^o(t+d|t,x(t)) &= y_x + \alpha(q^{-1},x(t))y(t) + \beta(q^{-1},x(t))u(t) &(2.3.11)\\
&= \Psi^T(t)\Xi(t) &(2.3.12)
\end{aligned}$$

where

$$x(t) = \begin{cases} \phi(t) & \text{for general quasi-ARX predictor} \\ \xi(t) & \text{for quasi-ARX predictor linear in } u(t) \end{cases}$$

and

$$\Psi(t) = [1, x^T(t)]^T$$
$$\Xi(t) = [y_x, \alpha_{0,t}, \cdots, \alpha_{n_y-1,t}, \beta_{0,t}, \cdots, \beta_{n_u+d-2,t}]^T.$$

2.4 Incorporating WNs Based on Clustering Partition Method

2.4.1 Incorporating WNs in Quasi-ARX Predictor

It is found from Eq.(2.3.11) that the nonlinear part is embedded into coefficients vector Ξ. To parameterize these coefficients, WNs are utilized with prior knowledge of data distribution, therefore some interpretable parameters of the model can be determined beforehand, and it is meaningful for nonlinear models to explain the structural characteristics of the system in real applications. To this end, the Q-ARX-WN predictor could be expressed explicitly by

$$\begin{aligned} y^o(t+d|t, x(t)) &= \Psi^T(t)(\Omega_0 + \sum_{j=1}^{M} \Omega_j \mathcal{N}(p_j, x(t)) \\ &= \Psi^T(t)\Omega_0 + \sum_{j=1}^{M} \Psi(t)\Omega_j \mathcal{N}(p_j, x(t)) + e(t) \end{aligned} \tag{2.4.1}$$

in which $\Xi(t) = \Omega_0 + \sum_{j=1}^{M} \Omega_j \mathcal{N}(p_j, x(t))$ is represented by a WN, where $\mathcal{N}(p_j, x(t))$ denotes the j-th wavelet neuron, and $\Omega_j = [\omega_{0j}, \cdots, \omega_{nj}]^T (j = 1, \cdots, M)$ is the connection matrix between input variables and the corresponding wavelet neurons. Here, p_j represents the parameter vector of the j-th wavelet neuron including dilation and translation parameter vector $D_{j\cdot} \in R^n$ and $T_{j\cdot} \in R^n$. n represents the length of input variables, and M is the size of wavelet neurons. Specifically, the j-th wavelet neuron could be expressed by

$$\mathcal{N}(p_j, x(t)) = \prod_{i=1}^{n} \psi\left(\frac{x_i(t) - T_{ji}}{D_{ji}}\right) \tag{2.4.2}$$

where ψ is the expression of wavelet function, and $x_i(t)$ is the i-th component of $x(t)$. Different from the parameters in neural networks (NNs), parameters in WNs are interpretable, and could be learned according to the non-uniform distribution of training data.

Remark : Parameters in the Q-ARX-WN model and predictor are categorized into linear parameters and nonlinear parameters. Concretely, in Eq.(2.4.1) parameters of wavelet neurons p_j are called nonlinear parameters, in contrast, Ω_j ($j = 0, 1, \cdots, M$) are called linear parameters and they are estimated by linear approaches.

2.4.2 Clustering Partition Method for Nonlinear Parameter Estimation

In order to determine the nonlinear parameters heuristically by using prior knowledge, the clustering partition method is used to extract data distribution information of the system. In the Q-ARX-WN prediction model, dilation ($D_{j\cdot}$) and translation ($T_{j\cdot}$) parameter vector for the j-th wavelet neuron need to be determined. Another important parameter is the size of wavelet neurons (M), which is aimed to construct the most parsimonious networks. It is expected that the input space partition is initialized automatically, where each subspace are well-covered by wavelet neurons, and the number of the partition needed could be determined according to data distribution.

Affinity propagation (AP) clustering algorithm is recently introduced for exemplar-based clustering method [59]. It is a new algorithm that takes as input measures of similarity $s(i, k)$ between pairs of data points i and k, and simultaneously considers all data points as potential exemplars. Real-valued messages are exchanged between data points until a high-quality set of exemplars and corresponding clusters gradually emerges. Because of its simplicity, general applicability, and good performance, we believe AP clustering will provide good performance for prior knowledge acquisition. The algorithm works by exchanging messages between the points until a stop condition is satisfied. There are two types of messages to be exchanged between data points. The responsibility $r(i, k)$, sent from data point i to candidate exemplar point k, reflects the accumulated evidence for how well-suited point k is to serve as the exemplar for point i, taking into account other potential exemplars for point i. The availability $a(i, k)$, sent from candidate exemplar point k to point i, reflects the accumulated evidence for how appropriate it would be for point i to choose point k as its exemplar, taking into account the support from other points that point k should be an exemplar. The availabilities are initialized to zero: $a(i, k) = 0$. Then, the parameters are computed and updated using the rules as follows:

$$r(i,k) \leftarrow s(i,k) - \max_{k' \text{ s.t. } k' \neq k} \{a(i,k') + s(i,k')\}$$

$$a(i,k) \leftarrow \min\left\{0, r(k,k) + \sum_{i' \text{ s.t. } i' \notin \{i,k\}} \max\{0, r(i',k)\}\right\}$$

$$a(k,k) \leftarrow \sum_{i' \text{ s.t. } i' \neq k} \max\{0, r(i',k)\}$$

In this way, it is believed that the input space can be partitioned automatically according to the data distribution information.

In the following, an heuristical guide is provided to determine the nonlinear parameters using the partition information. In the j-th cluster c_j, according to the reference [58], the domain contains value of the k-th component of input vectors is denoted by $[\min(x_k(c_j)), \max(x_k(c_j))]$, and translation and dilation parameters of the j-th wavelet neuron on the k-th input dimension are determined as

$$T_{jk} = 0.5[\max(x_k(c_j)) + \min(x_k(c_j))] \tag{2.4.3}$$

$$D_{jk} = 0.2[\max(x_k(c_j)) - \min(x_k(c_j))]. \tag{2.4.4}$$

These initializations are claimed to guarantee that the wavelets extend initially over the whole input domain [58]. However, some wavelet bases may be still redundant and should be eliminated, thus orthogonal least square (OLS) algorithm [18] is introduced to select the most important ones, and the size of basis could be determined by some well known approaches such as Akaike's information criterion (AIC) and Bayesian information criterion (BIC) [1]. This method has been successfully applied in research on WN [60, 16, 17].

2.4.3 Linear Parameter Estimation

When the nonlinear parameters are determined and fixed as Eq.(2.4.3) and (2.4.4), Eq.(2.4.1) could be described as

$$y^o(t+d|t, x(t)) = \Phi^T(t)\Theta + e(t) \tag{2.4.5}$$

in which

$$\Phi(t) = [\Psi^T(t), \mathcal{N}^T(x(t)) \otimes \Psi^T(t)] \tag{2.4.6}$$

$$\Theta = [\Omega_0^T, \Omega_1^T, \cdots, \Omega_M^T]^T \tag{2.4.7}$$

and

$$\mathcal{N}(x(t)) = [\mathcal{N}(p_1, x(t)), \cdots, \mathcal{N}(p_M, x(t))]^T$$

The symbol $\otimes$ in Eq.(2.4.6) denotes Kronecker production.

According to Eq.(2.4.5), it is obviously that the Q-ARX-WN predictor becomes linear-in-parameter and the corresponding parameters Θ can be estimated by using linear methods directly. Recursive least square (RLS) algorithm provides a simple approach for parameter estimation of the quasi-ARX models. Moreover, linear support vector regression (SVR) can also be applied, which is aimed to introduce robust performance, and the details can be find in Chapter 4.

2.5 Numerical Simulation

To show effectiveness of the clustering partition based Q-ARX-WN model for nonlinear system identification, real, mathematical and time series examples are tested. The first two examples are well-studied and have been identified by network models such as WN, NN, the quasi-ARX-neurofuzzy network (Q-ARX-NFN) and the quasi-ARX-neural network (Q-ARX-NN). The results of proposed model will be compared with them. Finally, an example of nonlinear adaptive control is given to show the advantage and usefulness of the Q-ARX-WN predictor, where a sudden change is happened on the system and tracked by adjusting linear parameters only using RLS algorithm. The metric to assess the accuracy of modeling is root mean squared error(RMSE):

$$\text{RMSE} = \sqrt{\frac{\sum_t (y(t) - \widehat{y}(t))^2}{N}}$$

where $\widehat{y}(t)$ is the prediction value of system output $y(t)$, and N is the number of regression vectors.

The used mother wavelet function in simulations is shown as

$$\psi(x) = (x^T x - n) e^{-\frac{1}{2} x^T x} \tag{2.5.1}$$

where n is the dimension of the input space. All the experiments are implemented on a personal computer with CPU of Intel Core2 Duo T9400(2.53GHz) and RAM of 3G by Matlab 7.6, and the linear parameters are estimated using linear SVR in system identification. Lib SVM toolbox version 2.91 [61] is applied, and ν-SVR is used in examples with default parameter setting. What's more, the value of d in Eq.(2.3.11) is set to 1 in simulations.

2.5.1 Identification of a Real System

System under Study

This is an example of modeling a hydraulic robot actuator, the position of a robot arm is controlled by a hydraulic actuator. The oil pressure in the actuator is controlled by the size of the valve opening through which the oil flows into the actuator. What we want to model is the dynamic relationship between the position of the valve $u(t)$ and the oil pressure $y(t)$. A sample of 1024 pairs of $\{y(t), u(t)\}$ was observed as shown in Fig.2.1. The data was divided into two equal parts, the first 512 samples were used as training data, and the rest were used to test the trained network. For the purpose of comparison, the regression vector is set as $\varphi(t) = [y(t-1), y(t-2), y(t-3), u(t-1), u(t-2)]^T$, and the number of wavelet neurons is selected as six, since the same setting has been found in references [33, 60] (Six wavelet neurons are used in WN based identification method).

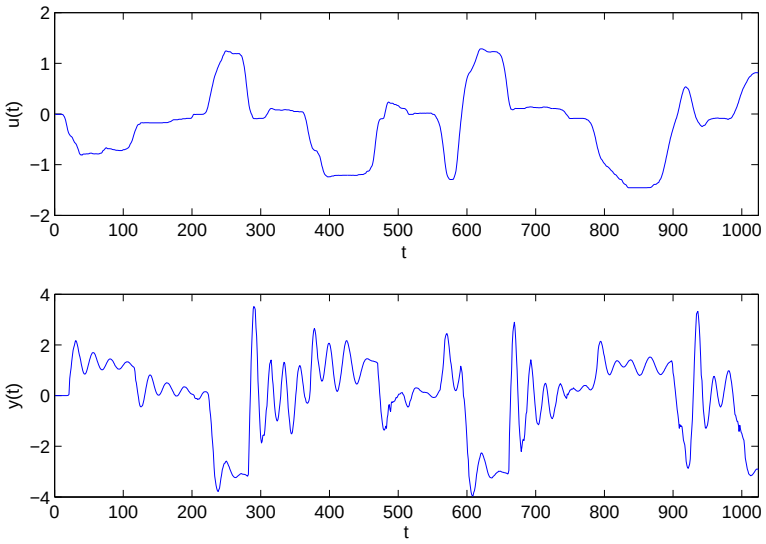


Figure 2.1: Measurement of system input and output.

Identification Results

To determine nonlinear parameters of the Q-ARX-WN model, AP clustering algorithm is implemented, and 11 clusters are generated automatically in the input space, thus 11 wavelet neurons are constructed according to Eq.(2.4.3) and (2.4.4) in each cluster. For simplicity and easy to compare with results from references [[16, 60]], OLS algorithm is applied to select the first six significant wavelet neurons. Therefore, there are 42 linear parameters to be estimated, where the biases of WN and the ARX-like macro-part are concerned. The simulation of the model on the test data is shown in Fig.2.2, where the solid line represents the real measurements and the dashed line represents the results of simulation. It gives a RMSE of 0.546.

It could also be found that if all the 11 clusters are used to construct wavelet neurons, the RMSE of the simulated model on test data could be reduced to as small as 0.467, which is shown in Fig.2.3.

Results Comparison

The simulation results from the clustering partition based Q-ARX-WN model are compared with the ones from linear ARX model, NN, WN, the Q-ARX-NFN and the Q-ARX-NN. The results are given in Tab.2.1.

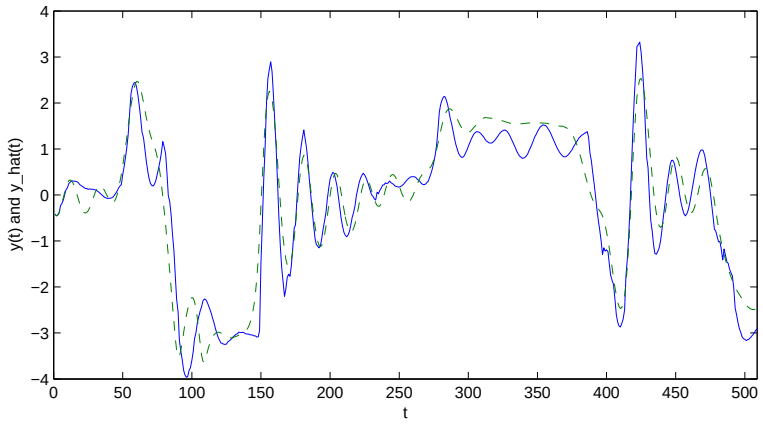


Figure 2.2: Simulation of the Q-ARX-WN model on test data for the real system.

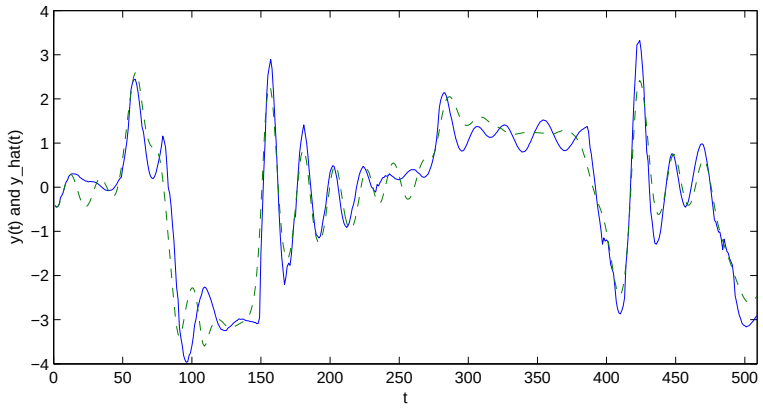


Figure 2.3: Simulation of the Q-ARX-WN model on test data with all the information contained.

Table 2.1: Experiment results and comparison for the real system.

Model	RMSE	number of parameters
ARX model	1.016	6
NN	0.617	71
WN	0.529	18
Q-ARX-NFN	0.545	85
Q-ARX-NN	0.657	65
Q-ARX-WN	0.546	54

In the Q-ARX-NFN model, it is difficult to use all the nodes according to the fuzzy rules because the number is as huge as 5^5 when five fuzzy sets are built on all the five input variables. Therefore,

the hint [33] should be used thus only 16 fuzzy rules are utilized for each input variable. In the Q-ARX-NN, a '5-5-5' network structure is incorporated and training by hierarchical algorithm proposed in [35]. However, the training for the embedded neural network is time-costing. When subtle changes have happened on the system, all the parameters of NN should be adjusted iteratively. In contrast, it can be found that the Q-ARX-WN model with prior knowledge is compatible to NN with 10 hidden units and WN with 6 wavelet neurons without losing the easy-to-use linear properties, where the number of parameters includes all the parameters in the model.

2.5.2 Identification of a Mathematical System

System Under Study

A benchmark mathematical problem is carried out to show the effectiveness of the clustering partition based Q-ARX-WN model. It is taken from Narendra [40], which contains rather strong nonlinearity. The description is given as:

$$y(t) = f[y(t-1), y(t-2), y(t-3), u(t-1), u(t-2)] + e(t)$$

where

$$f[x_1, x_2, x_3, x_4, x_5] = \frac{x_1 x_2 x_3 x_5 (x_3 - 1) + x_4}{1 + x_2^2 + x_3^2}$$

and $e(t) \in (0, 0.01)$ is white Gaussian noise. It is excited by 1000 random sequence with the amplitude between -1 and 1. Fig.2.4 shows the first 300 set of training data.

To test the obtained network models, a set of 800 input-output data is sampled as test data, and the input data is described as

$$u(t) = \begin{cases} \sin(2\pi t/250) & \text{if } t \leq 500 \\ 0.8\sin(2\pi t/250) + 0.2\sin(2\pi t/25) & \text{otherwise.} \end{cases}$$

Identification Results

To determine nonlinear parameters of the Q-ARX-WN model, AP clustering algorithm is implemented and 10 clusters are generated automatically in the input space, therefore, 10 corresponding wavelet neurons are constructed according to Eq.(2.4.3) and (2.4.4) in each cluster. Then OLS algorithm is applied and the most significant wavelet neurons could be selected. The RMSE of the simulated model with different wavelet neuron size is given in Tab.2.2.

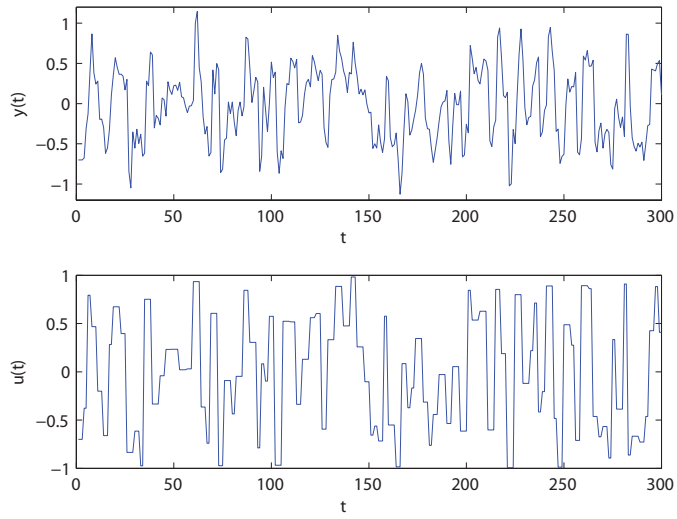


Figure 2.4: Training data for the mathematical system.

Table 2.2: RMSE with different wavelet neuron size.

Neuron size	2	3	4	5	6	7	8	9	10
RMSE	0.045	0.041	0.038	0.037	0.037	0.046	0.044	0.046	0.044

From the simulation, the best model on the test data is shown in Fig.2.5, where the solid line represents the real measurements and the dashed line represents the results of simulation. It contains 6 wavelet neurons and gives a RMSE of 0.037.

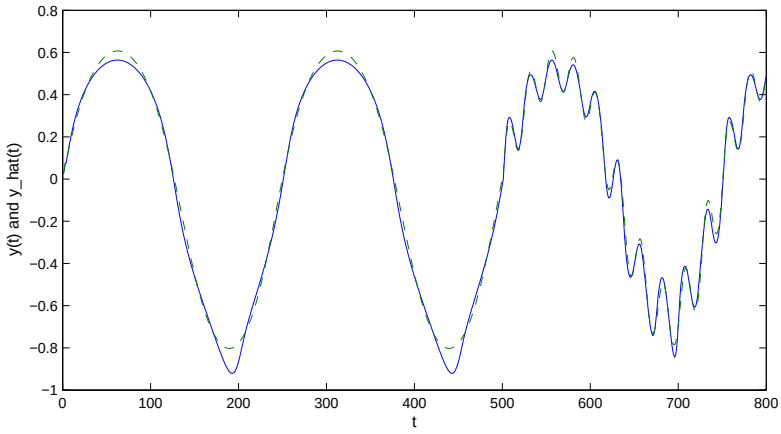


Figure 2.5: Simulation of the Q-ARX-WN model on test data for the mathematical system.

Results Comparison

The simulation results from the clustering partition based Q-ARX-WN model is compared with the ones from linear model, NN, WN, the Q-ARX-NFN and the Q-ARX-NN. The WN is also with six wavelet neurons contained, which is the same with the core-part of the Q-ARX-WN. The results are given in Tab.2.3.

Table 2.3: Experiment results and comparison for the mathematical system.

Model	RMSE	number of parameters
ARX model	0.087	6
NN	0.068	341
WN	0.050	18
Q-ARX-NFN	0.048	95
Q-ARX-NN	<0.01	246
Q-ARX-WN	0.037	54

It is found that the simulation RMSE from the Q-ARX-NN could be as small as below 0.01, when the number of iterations for training is big enough. However, it still suffers from long time training and non-linear-in-parameter property, and it is difficult to determine neurons size in hidden layer instructively. In the Q-ARX-NFN model, the hint is still needed and only part fuzzy rules are utilized for each input variable. A four-layer NN of $\mathcal{N}_{5,20,10,1}$ is used to identify the system, it contains 341 parameters, and the prediction is not as accurate as expected. Although ARX model and WN have simple structures, the simulation RMSE is not impressive compared with the proposed identification method.

2.5.3 Identification of a Time Series System

The time series prediction based on the chaotic Mackey-Glass differential equation is a standard benchmark problem for learning and generalization ability. This time series is generated from the following equation:

$$\frac{dx(t)}{dt} = \frac{ax(t-\tau)}{1+x^{10}(t-\tau)} - bx(t),$$

where $a = 0.2$, $b = 0.1$, and $\tau = 17$; the equation shows chaotic behavior. In keeping with most of the earlier work, we predict the $x(t+6)$ using the input variables $x(t)$, $x(t-6)$, $x(t-12)$, and $x(t-18)$, respectively. What's more, a white noise $e(t) \in (0, 0.01)$ is added on the system. 1000 sample points are used in our study, which is shown in Fig. 2.6. The first 500 data pairs of the series were used as training data, while the remaining 500 were used to test the model identified.

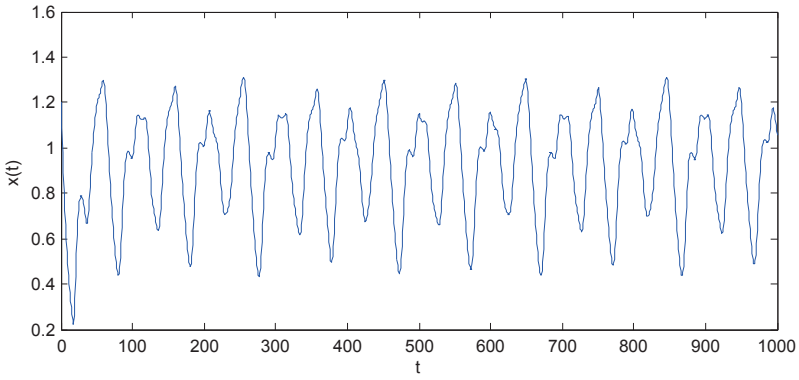


Figure 2.6: Actual time series data.

For identification using the clustering partition based Q-ARX-WN model, AP clustering algorithm is implemented, and 24 clusters are generated automatically in the input space, thus 24 wavelet neurons are constructed according to Eq.(2.4.3) and (2.4.4) in each cluster. The simulated result is shown in Fig.2.7, which gives a simulation RMSE of 0.047. In comparison, the simulation result from a linear ARX model is worse, which gives a RMSE of 0.19. Moreover, results from nonlinear methods such as NNs and fuzzy system which give RMSE of 0.02 and 0.049 [62], respectively, where the proposed method performs compatible with them, and is considered can solve time series prediction problem well.

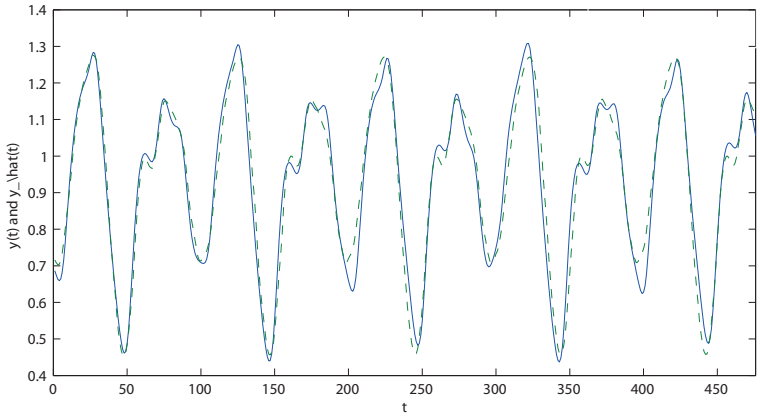


Figure 2.7: Simulation of the time series system.

2.5.4 Q-ARX-WN Predictor for Nonlinear Adaptive Control

In this example, an adaptive controller is designed using the Q-ARX-WN predictor, in which two advantages are mentioned. Firstly, the Q-ARX-WN predictor is linear in $u(t)$, therefore, controller could share parameters of the predictor, which is usually available limited in linear control system. Moreover, since the Q-ARX-WN predictor is linear-in-parameter when the nonlinear part is determined and fixed, nonlinear systems could be controlled online and with only linear parameters adjusted repeatedly, even sudden change is happened on the system.

For a minimum prediction error adaptive controller, consider a criterion function [34] as

$$J(t+d) = \frac{1}{2}[y(t+d) - y^*(t+d)]^2 + \frac{\lambda}{2}u(t)^2 \tag{2.5.2}$$

where λ is a weighting factor for the control input. Since the Q-ARX-WN predictor is linear for $u(t)$ in Eq.(2.3.10), differentiating Eq.(2.5.2) with respect to $u(t)$, the control law is obtained and expressed as

$$u(t) = \frac{\beta_{0,t}}{\beta_{0,t}^2 + \lambda}[\beta_{0,t} - \beta(q^{-1}, \xi(t)))q]u(t-1) + y^*(t+d) - \alpha(q^{-1}, \xi(t))y(t) - y_\xi \tag{2.5.3}$$

where $y^*(t)$ denotes reference output, and $\xi(t)$ is used to include an extra input variable instead of input variable $u(t)$. The adaptive control law is synthesized in the following way: Firstly the linear parameters Θ in Eq.(2.4.7) estimated in a recursive way using RLS algorithm; then calculate the control input $u(t)$ according to Eq.(2.5.3) by using $\hat{\Theta}$. Details of controller design and its convergence and stability could be find in the references [34, 35].

The numerical nonlinear system is similar with the mathematic example from Narendra [40], however, it is assumed that a sudden change has happened on the system when $t > 500$, hence the system is represented as

$$y(t) = f[y(t-1), y(t-2), y(t-3), u(t-1), u(t-2)] + e(t)$$

where

$$f[x_1, x_2, x_3, x_4, x_5] = \begin{cases} \frac{x_1 x_2 x_3 x_5 (x_3 - 1) + x_4}{1 + x_2^2 + x_3^2} & \text{if } t \leq 500 \\ \frac{x_1 x_2 x_3 x_5 (x_3 - 1) + x_4}{1 + 0.4 * x_2^2 + x_3^2} & \text{otherwise.} \end{cases}$$

$e(t) \in (0, 0.1)$ is a white noise. The desired output of system is assumed to be

$$y^*(t) = 0.6 * y^*(t-1) + r(t-1)$$

where $r(t) = sin(2\pi t/25) + sin(2\pi t/10)$.

To control this system, an adaptive controller is designed with $n_y = 3$ and $n_u = 2$, and λ in Eq.(2.5.3) is set to 0.001, and the size of wavelet neurons used is six. What should be mentioned is that the nonlinear parameters are fixed during the whole procedure, even change has happened on the system. For comparison, Fig.2.8 shows the result of controlled system output (solid lines), reference output (dashed lines) and control signal, in which, (a) and (c) are the results of the Q-ARX-WN predictor and linear predictor for system with sudden change, (b) is the result of the Q-ARX-WN predictor for system without change. It can be found that the Q-ARX-WN based adaptive control method performs better than a linear one, and it could cope with system sudden change well.

To show convergence properties of the adaptive controller, mean square errors (MSE) are calculated in a moving window for all the three cases tested above in Fig.2.9.

$$\text{MSE}(t) = \frac{1}{m} \sum_{k=t-m+1}^{t} (y(k) - y^*(k))^2$$

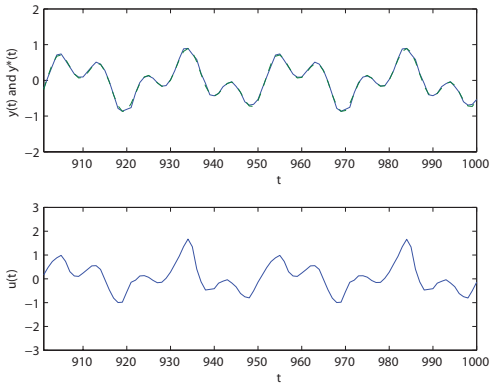
where m is the width of moving window and is chosen to be 100.

Obviously, the Q-ARX-WN predictor based adaptive control method outperforms the linear one. Moreover, when a sudden change happened on the system, the controller could track the reference signal quickly with only linear parameters adjusted.

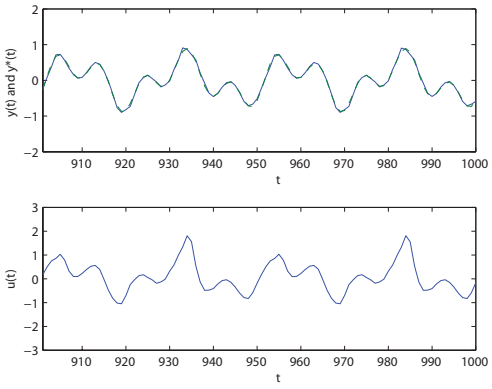
2.6 Discussions

1. This chapter has demonstrated effectiveness of the clustering partition based Q-ARX-WN model for nonlinear system identification and adaptive control, in which, WN is incorporated and the nonlinear parameters could be heuristically determined according to the distribution of training data. In fact, it is just considered as an example of the clustering partition based quasi-ARX models, and other basis function networks could also be embedded and implemented in the similar way, such as Radial Basis Function networks (RBFNs) in Chapter 4.

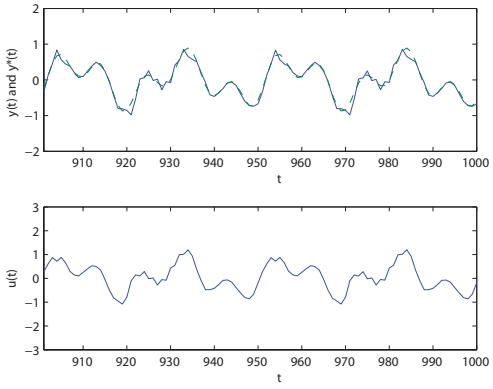
2. Not only the clustering partition method could be used for prior knowledge from WNs, other instructive methods are also practicable to guild the nonlinear part building. For example, the well-known wavelet frame theory derived WN [16] could also be initialized as the core-part of the Q-ARX-WN model. However, the simulation results are not as impressive as the ones in this chapter. The simulation RMSE on test data is 0.744 in the real data system, and 0.055 in the simulated data example, compared with 0.546 and 0.037 generated by the proposed method.



(a) Result of Q-ARX-WN predictor for system with a sudden change



(b) Result of Q-ARX-WN predictor for system without change



(c) Result of linear predictor for system with a sudden change

Figure 2.8: Result of nonlinear adaptive control: (a) and (c) are the results of the Q-ARX-WN predictor and linear predictor for system with sudden change, (b) is the result of the Q-ARX-WN predictor for system without change.

3. All the wavelet bases used in this research are extended widely enough to cover the interval of each input variable. Recalling Eq.(2.4.1), the Q-ARX-WN model could be explained as an ensemble model, in which the value of the j-th wavelet basis $\mathcal{N}(p_j, \varphi(t))$ could be seemed as the weight for the corresponding linear model $y_j(t) = \Psi^T(t)\Omega_j$. In fact, the quasi-ARX model could also be represented as a multi-local linear model, and each basis function could be used to cover just a local domain in the input space for interpolation, thus local linear property would be realized. Study on this issue will be given in Chapter 4.

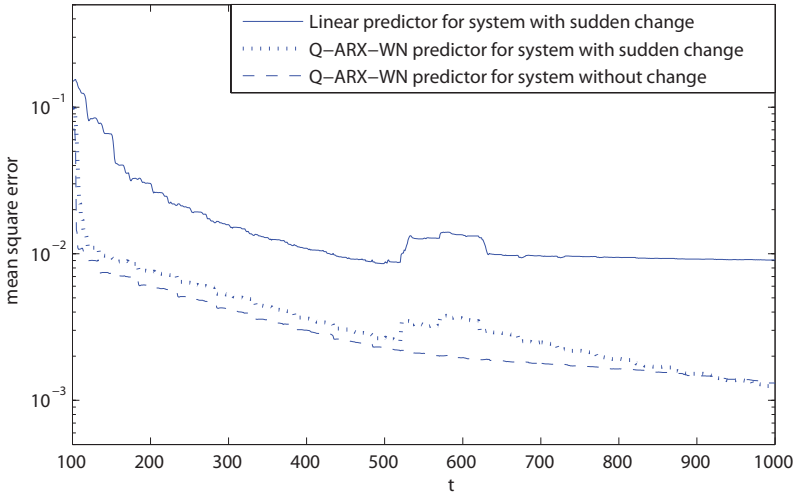


Figure 2.9: Mean square errors between $y(t)$ and $y^*(t)$ in a moving window.

2.7 Conclusions

In this chapter, the Q-ARX-WN model is identified by using clustering partition method, and there are mainly two contributions could be concluded. Firstly, the clustering partition method provides a rational guide for determining interpretable parameters of the model, which are interpretable and can be fixed during the identification. Secondly, the interpretable parameters in the Q-ARX-WN predictor are very useful for nonlinear adaptive control, since they can be fixed during the whole control process with only linear parameters adjusted, even a sudden change has happened on the system.

The effectiveness of the proposed identification method is tested by real, mathematical and time series system in simulations. The results show that the clustering partition based Q-ARX-WN model generates compatible simulation RMSE on the test data, which is compared with WN, NN, Q-ARX-NFN and Q-ARX-NN. Moreover, an adaptive control example is given to show usefulness and advantages of the Q-ARX-WN predictor.

Chapter 3

Identification of Quasi-ARX Models Using Grid Partition

3.1 Introduction

Neural networks (NNs) and neurofuzzy networks (NFNs) have shown highly sophisticated capability for nonlinear function approximation. However, from a user's point of view, these models are difficult to be used for establishing estimation and control scheme directly [63], since their parameters are basically viewed as vehicles for adjusting the fit to the data and rarely reflect physical considerations in the system [1]. Recently, the quasi-ARX neurofuzzy network (Q-ARX-NFN) model has been proved to have both universal approximation ability and easy-to-use linear properties in nonlinear system identification and control [34]. It owns an ARX-like linear structure, in which the coefficients are expressed by an NFN, rather than constants.

When NFN is incorporated into the linear structure, the grid partition method can be performed in a simple way to obtain prior knowledge for system identification [33]. Nevertheless, as a typical grid partition method, the fuzzy membership partition is implemented on each input variable of the NFN, thereby the Q-ARX-NFN model suffers from curse-of-dimensionality problem since the generated fuzzy rules increases exponentially with the input dimension. It may result in a complex model with a large number of parameters to be estimated, which is difficult to use and two problems may be incurred: the first one is the high computational cost to estimate such a big number of parameters; and the second one is the overly complex model structure may deteriorate the generalization capability [64].

Intuitively, heuristical hints have been applied to remove unimportant rules and simplify the model [33, 65, 34], in which the model input variables linear with the output, are considered

sufficient to capture system dynamics by the linear expression, thus are less important to be appeared in the incorporated neurofuzzy network (InNFN). Although the basis idea is ambiguous, it inspires us to reduce model complexity by incorporating only necessary input variables to the InNFN. In this chapter, linearity of model inputs to output is analyzed in terms of correlation analysis, which is considered as a factor of fitness function for a genetic algorithm (GA) [26] based fuzzy rule selection, where model complexity is trade-off against fitting performance by a modified Bayesian information criterion (BIC). As a popular model-based input selection method [66, 67, 14], GA provides a global optimal approach and has attracted much attention in these years. It is worthy to note that the computational burden of excessive parameters estimation is alleviated by using support vector regression (SVR), which is insensitive to the input dimension, and plays an important role to cope with curse-of-dimensionality of Q-ARX-NFN model. Details of this part will be given in Chapter 4.

This chapter begins with the problem formulation in Section 3.2, and a description of the grid based Q-ARX-NFN model is introduced in Section 3.3. In Section 3.4, a GA based fuzzy rule selection method is proposed for model generalization. Two benchmark problems are simulated in Section 3.5, discussions and conclusions are finally summarized.

3.2 Problem Formulation

Consider a single-input-single-output (SISO) nonlinear time-invariant system whose input-output dynamics is described as

$$\begin{aligned} y(t) &= g(\varphi(t)) + e(t) \\ \varphi(t) &= [y(t-1), \cdots, y(t-n_y), u(t-1), \cdots, u(t-n_u)]^T \end{aligned} \tag{3.2.1}$$

where $u(t) \in R$, $y(t) \in R$, $e(t) \in R$ are the system input, output and a stochastic noise of zero-mean at time t, n_u and n_y are unknown maximum delays of the input and output, respectively. $\varphi(t) \in R^n$ is the regression vector composed of delayed input-output data. n is the dimension of input variables, which equals to the sum of n_u and n_y. $g(\cdot)$:$R^{n=n_u+n_y} \rightarrow R$ is an unknown function (black-box) describing the dynamics of system under study, and the following two assumptions are introduced.

- Assumption 1: $g(\cdot)$ is a continuous function, but in a small region around $\varphi(t) = 0$, it is C^∞ continuous.

- Assumption 2: The input-output of system $u(t)$ and $y(t)$ are bounded, where the bounds are known a priori knowledge.

3.3 Q-ARX-NFN Model Based on Grid Partition

In the Q-ARX-NFN modeling, an ARX-like linear structure is constructed, which is useful for system analysis, and an NFN is embedded into the coefficients to parameterize the linear structure. The model can be described as following:

$$y(t) = \varphi^T(t)\theta(x(t)) + e(t) \tag{3.3.1}$$

$$\theta(x(t)) = \Omega_0 + \sum_{j=1}^{M} \Omega_j \mathcal{N}(p_j, x(t)) \tag{3.3.2}$$

where $\theta(x(t))$ is a state-dependant parameter of the ARX model, and it can be represented by the InNFN in Eq.(3.3.2). $\mathcal{N}$ denotes the neurofuzzy basis function, where $x(t) \in R^r, r \leq n$ contains the r selected input variables from system inputs. $\Omega_j = [\omega_{0j}, \cdots, \omega_{nj}]^T$ is a connection matrix between the input variables and the corresponding basis functions.

The neurofuzzy basis function corresponds to a set of linguistic fuzzy rules which can be represented in the following form:

R_j**:** IF x_1 is A_1^j and x_2 is A_2^j and $\cdots$ and x_r is A_r^j, THEN y is B^j

where $x_k,\ k = 1, 2, \cdots, r$ are input variables of the InNFN, and A_k^j denotes the fuzzy set of x_k. It is assumed that a Gaussian function [68, 10] is used as the membership function in adaptive fuzzy system [10] to form fuzzy rules, which can be represented as

$$\mu_{A_k^j}(x_k(t)) = \exp\left[-\frac{1}{2}\left(\frac{x_k(t) - \overline{x}_k^j}{\sigma_k^j}\right)^2\right]$$

where $\overline{x}_k^j, \sigma_j$ are center and width parameters of the Gaussian function. Then the neurofuzzy basis function is explicitly given as

$$\mathcal{N}(p_j, x(t)) = \frac{\prod_{k=1}^{r} \mu_{A_k^j}(x_k(t))}{\sum_{j=1}^{M}(\prod_{k=1}^{r} \mu_{A_k^j}(x_k(t)))}$$

where M means the total number of fuzzy rules, and p_j is the parameter vector given by

$$p_j = [\overline{x}_1^j, \cdots, \overline{x}_r^j, \sigma_1^j, \cdots, \sigma_r^j]^T.$$

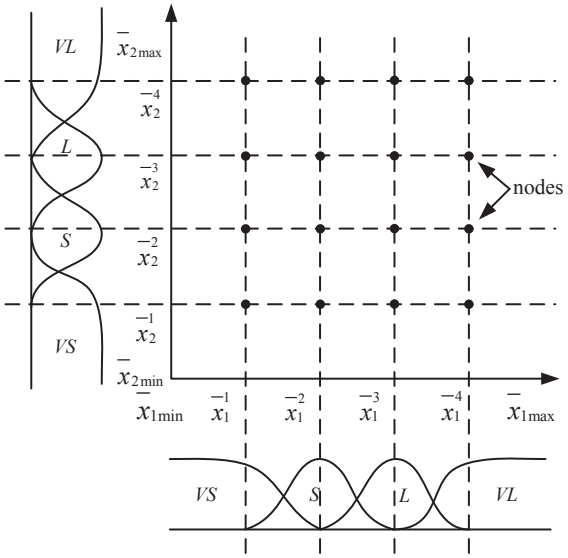


Figure 3.1: An example of fuzzy membership partition.

Usually the parameters are pre-determined based on prior knowledge using grid partition, for simplicity, the fuzzy membership partition is implemented in an uniform way on each dimension in this chapter. For instance, a two-dimension case is illustrated in Fig.3.1, where two inputs $\mathbf{x}_1$ and $\mathbf{x}_2$ are divided into even four parts, and the input space is covered by RBFs [10, 11] with center parameters of $\overline{\mathbf{x}} = [\overline{x}_1^1, \ldots, \overline{x}_1^4, \overline{x}_2^1, \ldots, \overline{x}_2^4]$. Correspondingly, the number of fuzzy basis functions is $M = 4 \times 4$. Generally speaking, M is increased exponentially with the dimension of input space r. When l fuzzy membership partitions are made on each input dimension, the number of fuzzy basis generated is

$$M = l^r.$$

thereby the number of parameters to be estimated from Eq.(3.3.2) is

$$N_p = (M + 1) \times n. \tag{3.3.3}$$

As a result, a large number of parameters need to be estimated, thus two problems should be concerned. For one thing, computational complexity for the parameter estimation is high. For another, over-fitting problem is easy to be caused.

3.4 GA Based Fuzzy Rule Selection

To alleviate computational cost of estimating such big number of parameters, SVR based approach is utilized which transformed the parameter estimation into a dual form of quadratic programming (QP) optimization, where the size of solution vector is only increased with number of observations, rather than input dimension. Details of this part will be shown in Chapter 4. However, the grid partition based Q-ARX-NFN model structure is still over-complex if all the fuzzy rules are included in the InNFN, which increased drastically with input dimensions. In recent years, many methods have been proposed to select input variables for nonlinear system, which can be categorized into model-free methods and model-based methods [66]. As a common used model-based method, GA can be used to search the appropriate and minimal input subset x for the InNFN, from all the inputs φ of the system.

3.4.1 Heuristical Hints for Fuzzy Rules Reduction

The prior knowledge from grid partition provides a big number of fuzzy rules for the core-part of the Q-ARX-NFN model. To reduce the fuzzy rule base, some hints have been introduced heuristically [33].

1. Hint A: If the system is linear with respect to x_i, the input variable is considered enough to capture dynamics by linear expression, thereby no need to be input into the InNFN, which is used to parameterize the linear structure.

2. Hint B: If no other useful information is available, the input variables $y(t-1)$ and $u(t-1)$ are preferred to be included in the InNFN only.

3. Hint C: If the role of fuzzy rules can be replaced by employing interpolation of nonlinear models, those rules may be removed from the inputs for the InNFN.

However, the above hints are only intuitive and ambiguous. In order to make the selection scheme less heuristically dependent, a GA based input selection method is proposed to select necessary inputs to generate fuzzy rules automatically.

3.4.2 Encoding for Input Selection

A simple binary-valued encoding method is applied. Since all the real inputs for the system is known as prior knowledge, the length of chromosome is set to the number of inputs. What's more,

each gene represents the corresponding input variable, and one-value means the input is selected for the InNFN, and zero-value denotes the corresponding input is not incorporated. For example, when the maximum delays of output and input are set as $n_y = 3$, $n_u = 2$, a chromosome with fix length of five is built, and the input subset of $y(t-1)$, $y(t-3)$, and $u(t-1)$ can be expressed by

1	0	1	1	0

3.4.3 Fitness Evaluation

Decode each individual into inputs for the InNFN, then one-step-ahead output prediction of the Q-ARX-NFN model $\hat{y}$ can be obtained by using the SVR-based approach. To evaluate the goodness of each selected input subset, on the one hand, trade-off between the model prediction accuracy and the number of selected inputs has been adopt by methods such as BIC [1]

$$\text{BIC} = N \cdot \ln(\frac{1}{N}\sum_{t=1}^{N}[y(t) - \hat{y}(t)]^2) + L\ln(N)$$

where N is the number of observations, and L denotes the number of selected input variables. On the other hand, since the inputs which are linear of the system output are considered unnecessary to the core-part of the Q-ARX-NFN model, correlation coefficients between the input variables and output vector are utilized to find those linear input variables, which is represented as

$$c_i = \left| \frac{\sum_{t=1}^{N}(x_i(t) - \overline{x_i})(y(t) - \overline{y})}{\sqrt{\sum_{t=1}^{N}(x_i(t) - \overline{x}_i)^2 \sum_{t=1}^{N}(y(t) - \overline{y})^2}} \right| \tag{3.4.1}$$

where $x_i(t)$ denotes the i-th component of $x(t)$, $\overline{x}_i$ and $\overline{y}$ are the mean of vector x_i and y respectively.

Consequently, the fitness value can be expressed by

$$\text{FIT} = N \cdot \ln(\frac{1}{N}\sum_{t=1}^{N}[y(t) - \hat{y}(t)]^2) + \lambda\sum_{i=1}^{n} c_i \cdot g_i \cdot \ln(N) \tag{3.4.2}$$

where n is the length of chromosome, and g_i is the binary value of gene in chromosome.

The first half part of Eq.(3.4.2) emphasizes on mean square error (MSE) of one-step-ahead output prediction, which is the same with BIC. In the latter part, not only the number of input variables, but also the linear correlationship of each selected one with the output is also considered.

λ is an appropriate weight factor to make the compromise. The smaller the fitness value is, the better is the input variable subset for the InNFN. It can be also inferred that

$$L = \lambda \sum_{i=1}^{n} c_i \cdot g_i$$

if $\lambda = 1$, $c_i = 1$. In other words, the proposed fitness function can be treated as a variation of BIC.

3.4.4 Selection, Crossover, Mutation

We use tournament selection. And for Crossover and Mutation, those standard ones for binary-valued GA as described in (Houck et. al., 1996) [69] are used.

3.5 Simulations and Results

Firstly, a numerical polynomial system, which is linear with $y(t-1)$ is tested to illustrate rationale of the proposed method. To show effectiveness of the GA based fuzzy rule selection scheme, a rational and real benchmark are further tested for identification of the Q-ARX-NFN model. The first case was studied by Narendra in 1990 [40], and the second one is an example of modeling a hydraulic robot actuator, taken from (Sjöberg *et al.*, 1995) [7]. Both the examples have been well identified by using the Q-ARX-NFN model, in which the input subsets for the InNFN have been selected heuristically, and parameters are estimated based on least square (LS) method (we call it Method 3 in the following). In the proposed method, GA is used to select necessary input variables for the InNFN to reduce fuzzy rule base. Moreover, as an input-dimension-insensitive algorithm, SVR has been used for parameter estimation, which transformed the original problem into a dual form depend on training data size. Therefore, four main identification approaches are compared to outline the importance and effectiveness of the proposed method.

- Method 1: Identification using the LS method without fuzzy rule selection scheme.
- Method 2: Identification using the SVR based approach without fuzzy rule selection scheme.
- Method 3: Identification using the LS method with fuzzy rule selected heuristically.
- Method 4: Identification using the SVR based approach with the GA based fuzzy rule selection scheme.

What's more, as a well-known black-box model which is insensitive to the input dimension, the NN is also used for comparison. The Matlab Neural Networks Toolbox is applied, and one-hidden-layer sigmoid NNs are constructed.

Simulations are implemented on a personal computer with CPU of Intel Core2 Duo T9400(2.53GHz) and RAM of 3GB by Matlab 7.6. Lib SVM toolbox version 2.91 [61] is applied for SVR implementation, and ν-SVR is used with default parameter setting.

3.5.1 Identification of a Polynomial System

System Under Study

The system is represented by a nonlinear polynomial model, and the input variable $y(t-1)$ is exactly linear with system output. The description is given as:

$$y(t) = 0.2y(t-1) + 0.3u(t-1)^2 - 0.5sin(u(t-1)y(t-2)^2) + e(t)$$

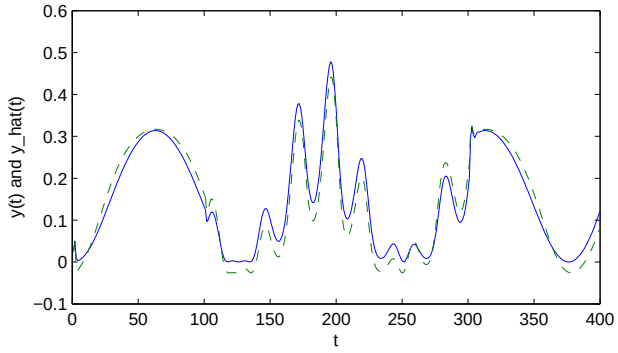
where $e(t) \in (0, 0.1)$ is white noise. The system is excited by 500 random sequence with the amplitude between -1 and 1, and the obtained model is tested by a set of 400 input-out test data, where the input is described as

$$u(t) = \begin{cases} 0.8\sin(2\pi t/250) + 0.2\sin(2\pi t/25) & \text{if } \ 101 \leq t < 300 \\ \sin(2\pi t/250) & \text{otherwise.} \end{cases}$$

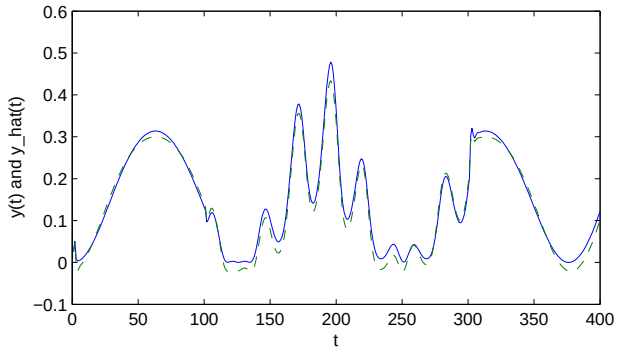
Experimental Results

The number of fuzzy membership partition on each input variable is set as $l = 3$. Therefore, when all the input variables are included into the InNFN, there are $M = 3^3$ fuzzy rules generated. Although it is a simple case, it is easy to illustrate the rationale of the proposed fuzzy rule selection scheme. Figure 3.2(a) shows the results of test when all the input variables are utilized in the InNFN, which gives a root mean square error (RMSE) of 0.027 for the simulated model. In contrast, excluding of the input $y(t-1)$, which is linear with the output and considered no need to be contained in the InNFN improves the model performance, which gives a RMSE of 0.017, and the simulated model is shown in Fig.3.2(b).

The results from the simulated model shows that the inputs which are linear with output are unnecessary to be contained in the InNFN for better generalization. Further comparison and analysis of the proposed method are achieved by the following examples.



(a) Result with all the system inputs in InNFN



(b) Result with only important system inputs in InNFN

Figure 3.2: Identification results to show rationale of the fuzzy rule selection scheme

3.5.2 Identification of a Rational System

System Under Study

The numerical system is represented by a nonlinear rational function, and it contains rather strong nonlinearity. The description is given as:

$$y(t) = f[y(t-1), y(t-2), y(t-3), u(t-1), u(t-2)] + e(t)$$

where

$$f[x_1, x_2, x_3, x_4, x_5] = \frac{x_1 x_2 x_3 x_5 (x_3 - 1) + x_4}{1 + x_2^2 + x_3^2}$$

and $e(t)$ is white noise. In this study, heavy noise is added for better understanding of the model generalization.

The system is excited by 1000 random sequence with the amplitude between -1 and 1. Figure 3.3 illustrates the first 300 set of training data with disturbance of $e(t) \in (0, 0.1)$. To test the obtained

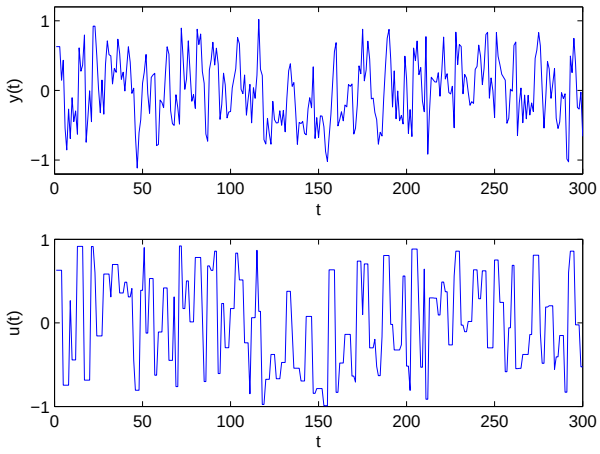


Figure 3.3: Training data for the rational system.

models by using the four identification approaches, a set of 800 input-output data is sampled as test data, and the input data is described as

$$u(t) = \begin{cases} \sin(2\pi t/250) & \text{if} \quad t < 500 \\ 0.8\sin(2\pi t/250) + 0.2\sin(2\pi t/25) & \text{otherwise.} \end{cases}$$

Experiment Setting

The number of fuzzy membership partition on each input variable is set as $l = 3$. Therefore, when all the input variables are included into the InNFN, there are $M = 3^5$ fuzzy rules generated. Moreover, parameter specifications of the GA based input selection are given in Tab.3.1. It should be

Table 3.1: Parameter setting for GA

GA Maximum generation	100
Population	20
Tournament size	2
Crossover possibility	0.7
Mutation possibility	0.3
λ	50

mentioned that λ is a weight factor for trade off by the trail-and-error, which is relatively insensitive to the algorithm and can be assigned in a large range in the experiment. In our simulations, the value of λ can be selected from an arbitrary integer between 50 and 200.

Results and Analysis

Firstly, simulations from the training data with disturbance of $e(t) \in (0, 0.3)$ are exemplified to give specific insight of effectiveness of the proposed method. The results and comparisons are shown in Tab.3.2. In which, the accuracy denotes model simulation RMSE on the test data, and the model

Table 3.2: Simulation results and comparison for the rational system

	Accuracy	Model Complexity	Inputs of InNFN
Method 1	Inf	1220(Number of parameters)	All
Method 2	0.247	557(Number of support vectors)	All
Method 3	0.067	140(Number of parameters)	y(t-1),y(t-2),y(t-3)
Method 4	0.048	516(Number of support vectors)	y(t-2),y(t-3)

Note: All $= \{y(t-1), y(t-2), y(t-3), u(t-1), u(t-2)\}$.

complexity is measured by the number of parameters to be computed. When LS method is utilized, it denotes to number of linear parameters to be estimated as Eq.(3.3.3) shown; when SVR based method is applied, the model complexity means number of support vectors. It is found that the model obtained by using the LS method without fuzzy rule selection scheme is failed to simulate the test data, because the number of parameters to be estimated is excessive and leads to serious over-fitting. In contrast, even though all the input variables are included, only 557 parameters are processed by the SVR based approach. The simulation of model on the test data is shown in Fig.3.4 (in order to show the results clearly, the bottom portions of the plot have been truncated), where the solid line shows the system true output and the dashed line denotes the simulated model output. It can be found that the result gives a RMSE of 0.247, which is not impressive and the simulated model is still over-fitted. Heuristically, the inputs can be selected for a simple model structure in Method 3, and the simulation on the test data is shown in Fig.3.5, which gives a RMSE of 0.067 by using LS method. In Method 4, the GA based selection scheme is applied and two variables $y(t-2)$ and $y(t-3)$ are selected automatically. The simulation result on the test data is shown in Fig.3.6, which gives a RMSE of 0.048, better than the one from heuristical selection method. For a more general comparison, the simulation of a one-hidden-layer with 20 hidden units, 5 input units and one output unit sigmoid NN is also given in Fig.3.7, which gives a RMSE of 0.064 on the test data.

From Fig.3.4-3.7 it is known that the proposed identification method outperforms others. In fact, the selected input subset $\{y(t-2), y(t-3)\}$ for the InNFN is considered reasonable, since the other three input variables are highly linear with the system output , and they are sufficient to capture system dynamics with linear expression. The correlation coefficient used in fitness evaluation for

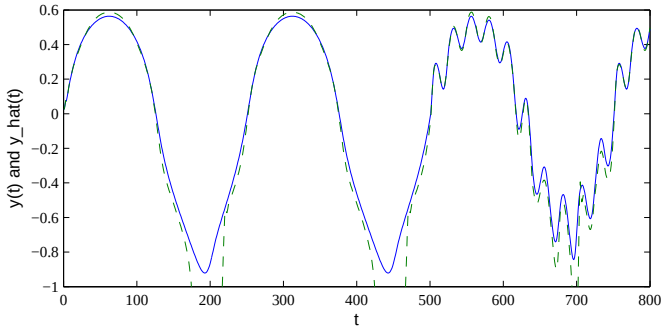


Figure 3.4: Identification result of the rational system by using Method 2

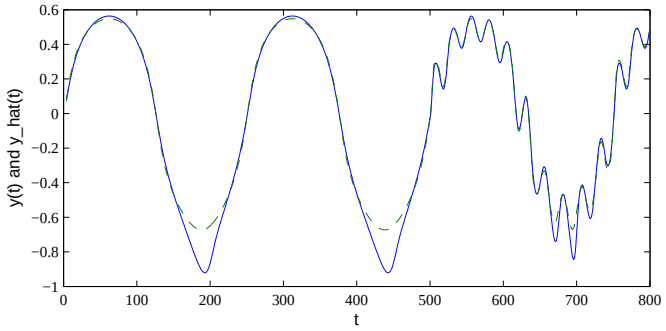


Figure 3.5: Identification result of the rational system by using Method 3

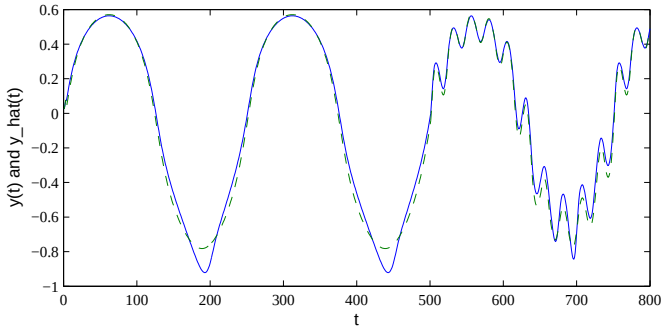


Figure 3.6: Identification result of the rational system by using Method 4

each input variable is shown in Tab.3.3.

In the following, the input subset from the GA based selection method will be compared with other suboptimal subsets. Five simulations under different variance of white disturbance are tested, and the RMSEs are shown in Tab.3.4.

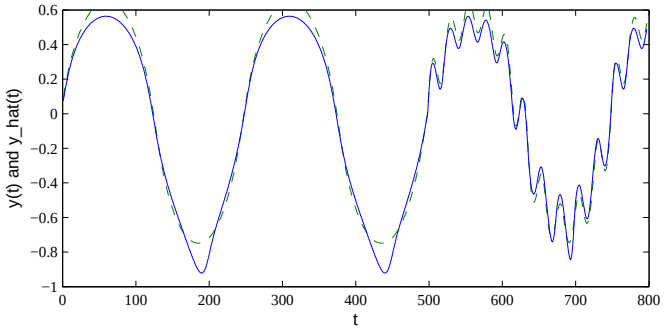


Figure 3.7: Identification result of the rational system by NN

Table 3.3: Correlation coefficient of each input variable to the output for the rational system

input variable	correlation coefficient
$y(t-1)$	0.404
$y(t-2)$	0.182
$y(t-3)$	0.058
$u(t-1)$	0.771
$u(t-2)$	0.498

Table 3.4: Comparison of suboptimal subsets under different variance of noise

Var	0.1	0.3	0.5	0.8	1.0	Mean value
S1	0.059	**0.048**	0.078	**0.082**	**0.072**	**0.068**
S2	0.057	0.367	0.096	0.093	0.096	0.142
S3	0.071	0.079	**0.069**	0.157	0.136	0.102
S4	0.057	0.166	0.114	**0.082**	0.112	0.106
All	0.060	0.247	0.108	0.102	0.081	0.120
NN	**0.052**	0.064	0.099	0.181	0.280	0.135

- S1 $= \{y(t-2), y(t-3)\}$ is the input subset selected by the GA.
- S2 $= \{y(t-1), y(t-2), y(t-3)\}$ is a suboptimal subset which was also generated heuristically in the reference.
- S3 $= \{y(t-1), u(t-1))\}$ and Subset4$= \{y(t-2), y(t-3), u(t-2)\}$ are suboptimal subsets which are found in all the five tests.
- All $= \{y(t-1), y(t-2), y(t-3), u(t-1), u(t-2)\}$.
- NN denotes system identification by using NN.

It is found that the input subset selected by the GA commonly outperforms others, and the proposed identification method outperforms the NN when variance of disturbance on the system

is increased. It should be mentioned that although the NN may give compatible or better performance when its structure and learning rate is optimized, it has neither useful interpretations of its parameters nor the structure favorable to applications such as controller design [35].

3.5.3 Identification of a Real System

System Under Study

This is an example of modeling a hydraulic robot actuator. The position of a robot arm is controlled by a hydraulic actuator. The oil pressure in the actuator is controlled by the size of the valve opening through which the oil flows into the actuator. The position of the robot arm is then a function of the oil pressure. Let us denote by $u(t)$ and $y(t)$ the position of the valve and the oil pressure at time t, respectively. A sample of 1024 pairs of $\{y(t), u(t)\}$ was registered as shown in Fig.3.8. It is divided into two equal parts for training and testing, respectively.

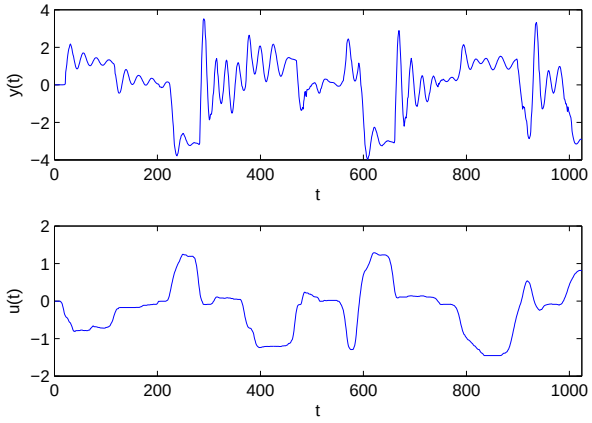


Figure 3.8: Measurement of system input and output.

Experiment Setting

Since the system is rather complex and with strong nonlinearity, the number of fuzzy membership partition on each input variable is set as $l = 5$. Therefore, $M = 5^5$ fuzzy rules are generated if all the input variables are included into the InNFN. The GA parameters are set as the ones in Tab.3.1 except the weighting factor λ, which is still insensitive and simply set to 20 in this example.

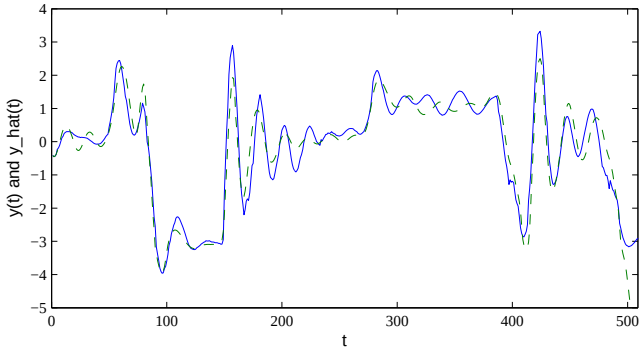


Figure 3.9: Identification result of the real system by using Method 2

Results and Analysis

Simulation results and comparisons are given in Tab.3.5. There are 15630 linear parameters con-

Table 3.5: Simulation results and comparison for the real system

	Accuracy	Model Complexity	Inputs of InNFN
Method 1	NaN	15630(Number of parameters)	All
Method 2	0.598	302(Number of support vectors)	All
Method 3	0.606	130(Number of parameters)	y(t-1),u(t-1)
Method 4	0.552	273(Number of support vectors)	y(t-2),u(t-2)

tained if all the five input variables are included in the InNFN, which is too complex for the LS method and out of memory by our PC. Although only two input variables are selected heuristically in Method 3, the performance is not impressive, and the simulation result on the test data is given in Fig.3.10, where the solid line shows the system true output and the dashed line denotes the simulated model output. In contrast, when the SVR based identification method is applied, the problem becomes tractable even with all the fuzzy rules considered, since only 302 support vectors are utilized for computation. The simulation result is shown in Fig.3.9, which gives a RMSE of 0.598. In Method 4, the input subset $\{y(t-2), u(t-2)\}$ is obtained by using the GA based selection scheme. The simulated model on the test data is given in Fig.3.11, and the RMSE is 0.552, better than the one without fuzzy rule selection scheme. What's more, a one-hidden-layer NN with 20 hidden nodes is constructed. The simulation of the model on the test data is shown in Fig.3.12, which gives a RMSE of 0.569, and we can see that the proposed method has a compatible performance.

However, it is found that the selected subset may not be the optimal, because the linear correlation coefficients for all the inputs to output are similar, hence the difference between them are

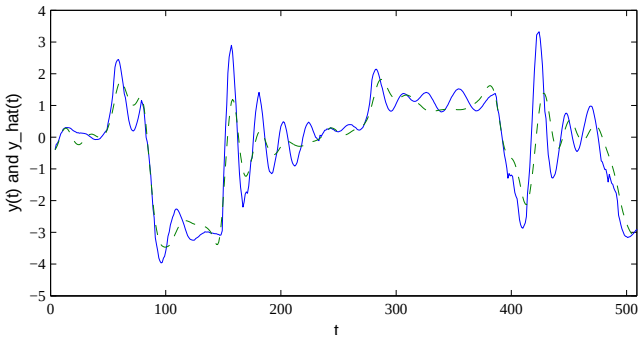


Figure 3.10: Identification result of the real system by using Method 3

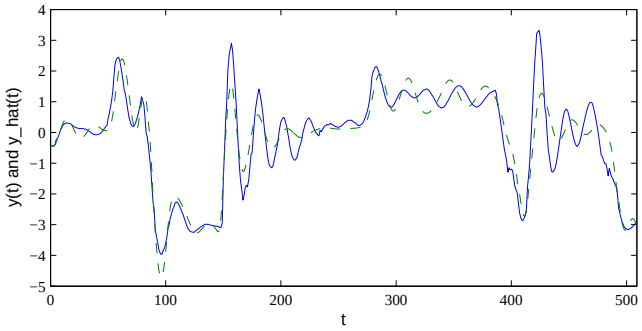


Figure 3.11: Identification result of the real system by using Method 4

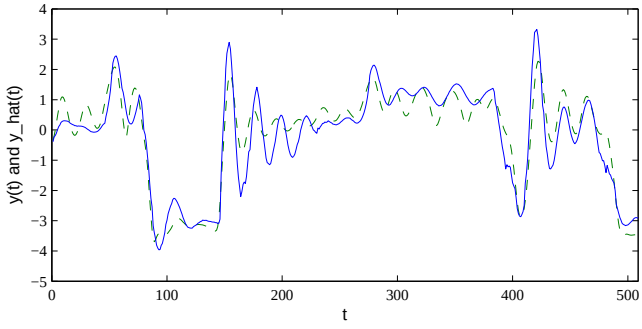


Figure 3.12: Identification result of the real system by NN

not obvious enough for evaluation. The correlation coefficient of each input to the output is are shown in Tab.3.6. Therefore, we give different weighted factors to the input variables according to

Table 3.6: Correlation coefficient of each input variable to the output for the real system

input variable	correlation coefficient
$y(t-1)$	0.9892
$y(t-2)$	0.9599
$y(t-3)$	0.9148
$u(t-1)$	0.8073
$u(t-2)$	0.8071

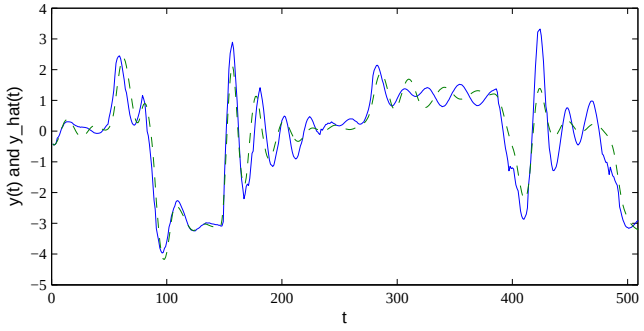


Figure 3.13: Simulation of identified model with uneven weighted factor for input variables

the value of correlation coefficients, thus Eq.(3.4.2) can be rewritten as

$$\text{FIT} = N \cdot \ln(\frac{1}{N}\sum_{t=1}^{N}[y(t) - \hat{y}(t)]^2) + \sum_{i=1}^{n}\lambda_i \cdot c_i \cdot g_i \cdot \ln(N)$$

where λ_i gives uneven compromise to input variables. In this example, $\lambda = [\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5]^T = [30, 20, 10, 20, 10]^T$ is set, which corresponds to $\varphi(t)$. The input subset $\{y(t-3), u(t-2)\}$ is obtained, and the simulation of the model on the test data can be found in Fig.3.13, which has a better prediction performance with the RMSE of 0.501.

3.6 Discussions and Conclusions

Compared with conventional nonlinear black-box models such as NNs and NFNs, the Q-ARX-NFN model is equipped with a linear structure and has some useful linear properties. At the same time, the grid partition provides useful prior knowledge for nonlinear parameter determination. Although the model is suffered from the curse-of-dimensionality problem, from the simulations it is known that the GA based fuzzy rule selection scheme can improve the model generalization. Comparing the simulation results from state-of-art algorithms such as NN, system identification using the grid partition based Q-ARX-NFN model gives a compatible performance.

What's more, the value of λ is fixed by trail-and-error method in this research. It is believed that a reasonable estimation of λ is important, which can improve the proposed algorithm as shown in the simulation of the real system example. It should be mentioned that the linearity of model inputs to the output is analyzed and included in the fitness evaluation. It is measured by correlation coefficient in this method, which provides an effective approach to find the inputs linear with the system. However, correlation coefficient is not good to represent variable linearity all the time, and some other efficient methods for the linearity evaluation will be studied in the future.

In numerical simulations, rationale of the proposed selection method to simplify model structure is testified by a simple example. Two benchmark cases are given and shown that although SVR based identification can alleviate computational burden to estimate big number of parameters, the model structure is still over-complex. GA based selection method can eliminate redundant fuzzy rules effectively with the consideration of linear correlation analysis of each input variable. In the next chapter, the SVR based identification method, which plays an important part in parameter estimation of the quasi-ARX model, will be discussed elaborately.

Chapter 4

Identification of Quasi-ARX Models Using Kernel Learning Approach

4.1 Introduction

In recent years, kernel methods provide a framework for tackling some profound issues in machine learning theory. At the same time, successful applications have demonstrated that support vector machines (SVMs) not only have a more solid foundation than artificial neural networks, but are able to serve as a replacement for neural networks that perform as well or better, in a wide variety of fields [23]. Support vector regression (SVR) is based on Vapnik's ε-insensitive loss function and structural risk minimization [25], which was subsequently proposed as the SVM implementation for regression and function approximation [49]. It has several appealing properties for black-box identification, thus has been studied and applied successfully to identification problems, including linear ARMA models [70, 71], nonlinear models [72, 22], and state-dependent parameter models [48] as well. Among which, the kernel technique plays an important role, which perform nonlinear mapping to a high-dimensional feature space implicitly by replacing the inner product with a positive definite kernel function. It is well known that the major task of the SVM approach lies in the selection of its kernel, and choosing different kernel functions will produce different SVMs [20, 50]. In the literature, there exist polynomial SVMs, radial basis function (RBF) SVMs, two-layer neural network SVMs, and so on. Moreover, the composite kernel which combines different types of kernel functions has attracted much attention these years for the flexible kernel function design [73, 74, 75].

However, from the view of applications, such kernel trick with the implicit kernel mapping is not always good because:

1. It is difficult to choose appropriate kernel from a limited number of existed kernel functions

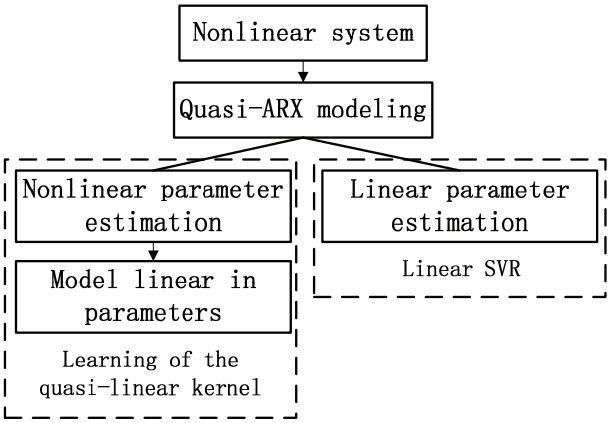


Figure 4.1: Basic idea of the proposed method

for some certain applications. Although the composite kernel functions satisfy the Mercer's condition and lead to many choices of hybrid kernels, it is not easy work which depends on the wit of the users and their understanding of the processing data.

2. The implicit nonlinear kernel mapping may face potential over-fitting problems for some complex and noised learning task. For instance, in functional genomics and microarray data some tasks are nonlinear in nature, with characteristics of high noise, and large number of input features compared with the relatively small number of training examples. It is found that the SVMs with implicit nonlinear kernel functions such as Gaussian kernel are severely over-fitting at times, thus can not obtain good results and even perform less than the linear model [76, 52, 53].

The purpose of this chapter is to identify the quasi-ARX model using kernel learning method, where SVR with a composite kernel function is proposed with the help of prior knowledge. The implementation is shown in Fig.4.1, where parameters of the quasi-ARX model are divided into nonlinear and linear part. The nonlinear parameters can be learnt and fixed by prior knowledge as shown in Chapter 2 and 3, which is devoted to the interpretable interpolation of local linear model, and details of this part can be found in Section 4.3.2. When nonlinear parameters are determined and fixed, the model is, in fact, converted linear-in-parameter, where the process is rightly in the sense of explicit kernel mapping. Therefore, the linear parameters can be solved by using SVR in the reproduced feature space. In this way, the robust performance mentioned in Chapter 2 is introduced, and computational cost for linear parameter estimation becomes independent to the input dimension, which is useful to solve curse-of-dimensionality problem of grid partition methods.

Moreover, the explicit kernel mapping provides a physically meaningful approach to reproduce the feature space, which is interpretable and can be obtained from prior knowledge by clustering partition and grid partition method (see Chapter 2 and 3). The associated kernel function is called quasi-linear kernel, and it is categorized into composite kernels with a novel construction method. Hopefully, the quasi-linear kernel can encourage a wider application of kernel techniques in the real world.

This chapter begins with the problem formulation in Section 4.2, and a description of kernel learning for the quasi-ARX model identification is given in Section 4.3. In Section 4.4, SVR based identification method with the quasi-linear kernel is proposed. In Section 4.5, numerical and real simulations and two applications to microarray missing value imputation are simulated to show the effectiveness of the proposed method, discussions and conclusions are summarized at last.

4.2 Problem Formulation

Consider a single-input-single-output (SISO) nonlinear time-invariant system whose input-output dynamics is described as

$$\begin{aligned} y(t) &= g(\varphi(t)) + e(t) \\ \varphi(t) &= [y(t-1), \cdots, y(t-n_y), u(t-1), \cdots, u(t-n_u)]^T \end{aligned} \tag{4.2.1}$$

where $u(t) \in R$, $y(t) \in R$, $e(t) \in R$ are the system input, output and a stochastic noise of zero-mean at time t, n_u and n_y are unknown maximum delays of the input and output, respectively. $\varphi(t) \in R^n$ is the regression vector composed of delayed input-output data. n is the dimension of input variables, which equals to the sum of n_u and n_y. $g(\cdot)$:$R^{n=n_u+n_y} \rightarrow R$ is an unknown function (black-box) describing the dynamics of system under study, and the following two assumptions are introduced.

- Assumption 1: $g(\cdot)$ is a continuous function, but in a small region around $\varphi(t) = 0$, it is C^∞ continuous.

- Assumption 2: The input-output of system $u(t)$ and $y(t)$ are bounded, where the bounds are known a priori knowledge.

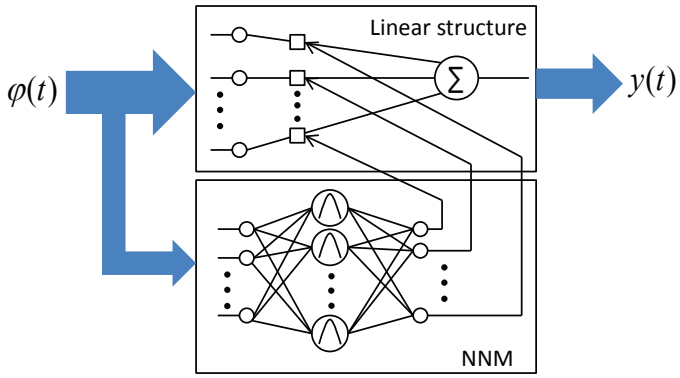


Figure 4.2: Diagram of the quasi-ARX modeling

4.3 Kernel Learning Based on Quasi-ARX Modeling

4.3.1 Quasi-ARX Modeling

The quasi-ARX modeling method is proposed for combining both the easy-to-use linear properties and universal approximation ability. As Fig.4.2 illustrated, an ARX-like linear structure is constructed, which makes the model easy to use especially in nonlinear controller design [35]. Moreover, it incorporates an nonlinear nonparametric model (NNM) to make the model flexible and provide much insight into system dynamics, in which some parameters are interpretable thus can be learnt and fixed by using clustering partition or grid partition method.

The model is generated by performing Taylor expansion to the unknown nonlinear function $g(\varphi(t))$ around the region $\varphi(t) = 0$. Since $g(\cdot)$ is assumed to be continuously differentiable, the derivative $g^{(i)}(0)(i = 1, 2, \cdots)$ exists. Then ignoring $g(0)$ for simplicity and a regression form of the system described in (3.1) could be generated as

$$y(t) = \varphi^T(t)\theta(\varphi(t)) + e(t) \tag{4.3.1}$$

where

$$\begin{aligned} \theta(\varphi(t)) &= (g'(0) + \frac{1}{2}\varphi^T(t)g''(0) + \cdots) \\ &= [a_{1,t}, \cdots, a_{n_y,t}, b_{0,t}, \cdots, b_{n_u-1,t}]^T. \end{aligned} \tag{4.3.2}$$

Here, the coefficients $a_{i,t} = a_i(\varphi(t))$ and $b_{i,t} = b_i(\varphi(t))$ are nonlinear functions of $\varphi(t)$, thus can be represented by NNMs in Eq.(4.3.3)

$$\theta(\varphi(t)) = \Omega_0 + \sum_{j=1}^{M} \Omega_j \mathcal{N}(p_j, \varphi(t)) \tag{4.3.3}$$

where p_j is parameter vector of the j-th basis function of the NNM $\mathcal{N}$, such as the center (μ) and width (σ) parameters in radial basis function network (RBFN), or fuzzy membership partition parameters in neurofuzzy network (NFN). M denotes the number of basis functions contained in the NNM, and the parameters p_j and M are called nonlinear parameters thereby. $\Omega_j = [\omega_{1j}, \cdots, \omega_{nj}]^T$ is a connection matrix between the input variables and the associated basis functions. According to Eq.(4.3.1) and (4.3.3), a compact representation is given as

$$y(t) = \sum_{j=0}^{M} \varphi^T(t)\Omega_j \mathcal{N}(p_j, \varphi(t)) + e(t) \tag{4.3.4}$$

in which it is defined

$$\mathcal{N}(p_0, \varphi(t)) = 1.$$

4.3.2 Explicit Nonlinear Mapping

From Eq.(4.3.4), the quasi-ARX modeling can be explained as a multi-local linear model with interpolation, which is expressed by an unknown NNM. It is known that some parameters in the incorporated NNMs are interpretable, and can be determined by means of prior knowledge from the system. Two approaches are proposed for the prior knowledge acquisition and parameter determination, which have be elaborated in Chapter 2 and 3. In this chapter, the RBFNs [13] are exemplified to represent $\mathcal{N}(p_j, \varphi(t))$ using clustering partition based prior knowledge, which can be explained as an interpolation for associated local linear model. This idea can be illustrated in Fig.4.3 for a one-dimensional example.

The parameter vector p_j (μ_j and σ_j in RBFs) are nonlinear parameters to be estimated, which are determined in a clustering partition way. In multi-dimensional situations, the input space can be partitioned by using clustering algorithms, and each sub-region is covered by a corresponding RBF, which can be represented as

$$\mathcal{N}(p_j, \varphi(t)) = \exp\left[- \frac{\left\| \varphi(t) - \mu_j \right\|^2}{\lambda \sigma_j^2} \right]. \tag{4.3.5}$$

Heuristically the j-th basis function $\mathcal{N}(p_j, \varphi(t))$ is formulated at the center of μ_j with width of σ_j, which are set as the center and radius of the j-th data cluster, and λ is a scale parameter.

After estimation and fixing of nonlinear parameters, Eq.(4.3.4) can be rewritten in a linear-in-parameter way as

$$y(t) = \Phi^T(t)\Theta + e(t) \tag{4.3.6}$$

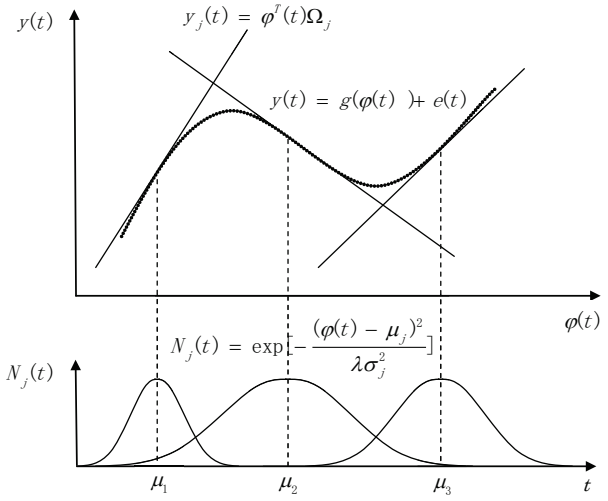


Figure 4.3: Multi-local linear model with interpolation for the quasi-ARX modeling

where

$$\Phi(t) = [\varphi^T(t), \mathcal{N}_1(t)\varphi^T(t), \cdots, \mathcal{N}_M(t)\varphi^T(t)]^T \tag{4.3.7}$$

$$\Theta = [\Omega_0^T, \Omega_1^T, \cdots, \Omega_M^T]^T \tag{4.3.8}$$

$$\mathcal{N}_j(t) \triangleq \mathcal{N}(p_j, \varphi(t)), \quad j = 1, \cdots, M.$$

Therefore, the nonlinear system identification is reduced to a linear regression problem with respect to $\Phi(t)$, which is rightly in the sense of kernel mapping. Θ are thereby called linear parameters.

Remark 1: It should be mentioned that the nonlinear input data becomes linear in a high-dimensional feature space in terms of nonlinear parameter learning. In other words, explicit kernel mapping is learnt to reproduce an appropriate feature space using the quasi-ARX model.

4.4 Identification Using SVR with Quasi-linear Kernel

In the reproduced feature space, the linear regression is estimated by structural risk minimization principal as

$$\min \mathcal{J} = \frac{1}{2}\Theta^T\Theta + C\sum_{t=1}^{N}(\xi_t + \xi_t^*) \tag{4.4.1}$$

subject to:

$$\begin{cases} y(t) - \Phi^T(t)\Theta \leq \epsilon + \xi_t^* \\ -y(t) + \Phi^T(t)\Theta \leq \epsilon + \xi_t, \end{cases}$$

where N is the number of observations, and ξ_t,$\xi_t^* \geq 0$ are slack variables. C is a non-negative weight to determine how much prediction errors are penalized, which exceed the threshold value ϵ. The solution can be turned to find a saddle point of the associated Lagrange function

$$\begin{aligned} & \mathcal{L}(\Theta, \xi_t, \xi_t^*, \alpha, \alpha^*, \beta, \beta^*) \\ = & \frac{1}{2}\Theta^T\Theta + C\sum_{t=1}^{N}(\xi_t + \xi_t^*) + \sum_{t=1}^{N}\alpha_t(y(t) - \Phi^T(t)\Theta - \epsilon - \xi_t) \\ & + \sum_{t=1}^{N}\alpha_t^*(-y(t) + \Phi^T(t)\Theta - \epsilon - \xi_t^*) - \sum_{t=1}^{N}(\beta_t\xi_t + \beta_t^*\xi_t^*). \end{aligned} \tag{4.4.2}$$

Then the saddle point could be acquired by minimization of $\mathcal{L}$ with respect to Θ, ξ_t^* and ξ_t

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \Theta} = 0 \Rightarrow \Theta = \sum_{t=1}^{N}(\alpha_t - \alpha_t^*)\Phi(t) \\ \frac{\partial \mathcal{L}}{\partial \xi_t^*} = 0 \Rightarrow \beta_t^* = C - \alpha_t^* \\ \frac{\partial \mathcal{L}}{\partial \xi_t} = 0 \Rightarrow \beta_t = C - \alpha_t \end{aligned} \tag{4.4.3}$$

Thus, one can convert the primal problem Eq.(4.4.1) into a equivalent dual problem as

$$\begin{aligned} \max \mathcal{W}(\alpha, \alpha^*) \ = \ & -\frac{1}{2}\sum_{t,k=1}^{N}(\alpha_t - \alpha_t^*)(\alpha_k - \alpha_k^*)\Phi^T(t)\Phi(k) \\ & + \sum_{t=1}^{N}(\alpha_t - \alpha_t^*)y(t) - \epsilon\sum_{t=1}^{N}(\alpha_t + \alpha_t^*) \end{aligned} \tag{4.4.4}$$

subject to:

$$\sum_{t=1}^{N}(\alpha_t - \alpha_t^*) = 0, \quad \alpha_t, \alpha_t^* \in [0, C].$$

With solutions α_t and α_t^* obtained, the parameter Θ from Eq.(4.4.3) is estimated in a dual form and represented as

$$\hat{\Theta} = \sum_{t=1}^{N}(\hat{\alpha}_t - \hat{\alpha}_t^*)\Phi(t).$$

In this way, two advantages can be concluded.

Remark 2: The robust performance for parameter estimation is introduce since the structure risk minimization of SVR;

Remark 3: There is no need to calculate linear parameters Θ directly. Instead, it becomes a dual form of quadratic optimization, which is represented using α_t and α_t^* and depended on size of training data. It is useful to alleviate computational cost when the the model is suffered from curse-of-dimensionality of grid partition method.

What's more, the nonlinear system from Eq.(4.2.1) can be identified with the quasi-linear kernel

$$\hat{y}(t) = \sum_{t'=1}^{N} (\hat{\alpha}_{t'} - \hat{\alpha}_{t'}^*) \mathcal{K}(t, t') \tag{4.4.5}$$

in which the kernel function is defined as inner product of the explicit nonlinear mapping

$$\mathcal{K}(t, t') = \Phi^T(t)\Phi(t') = \mathcal{K}_{\mathcal{L}}(t, t') \sum_{i=0}^{M} \mathcal{N}_i(t)\mathcal{N}_i(t'). \tag{4.4.6}$$

From Eq.(4.4.5) it is known identification of the quasi-ARX model is transformed to an SVR with the quasi-linear kernel, which is learnt from an explicit and physically meaningful nonlinear mapping. In Eq.(4.4.6), the quasi-linear kernel is explained as a composite kernel, in which $\mathcal{K}_{\mathcal{L}}(t, t') = \varphi^T(t)\varphi(t')$ is a linear kernel function, and $\mathcal{N}_i(t)$ is the result of the i-th basis function of NNMs. There are some remarks for explaining such quasi-linear kernel.

Remark 4: The quasi-linear kernel in Eq.(4.4.6) is satisfied with Mercer's condition, since it is defined as an inner product, and can be expanded in the form of the product of a linear kernel and the sum of Mercer kernels.

Remark 5: The quasi-linear kernel is named after two folds. Firstly, it is derived from the quasi-ARX modeling method. Secondly, the nonlinearity of the kernel can fill the gap between linear and nonlinear kernel functions by adjusting the value of M as shown in Fig.4.4. The appropri-

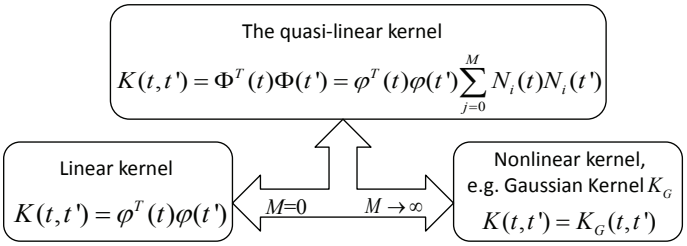


Figure 4.4: Nonlinearity of the quasi-linear kernel

ate nonlinearity can be achieved heuristically or by learning method. Details can be found in the simulations.

4.5 Simulations and Results

In this section, we will apply SVR with the quasi-linear kernel to identify numerical and real systems and solve a missing value imputation problem in microarray datasets. In the first numerical system, we will investigate effectiveness of the quasi-linear kernel with increasing the mapping nonlinearity. Then a contaminated system is tested to show robust performance of SVR for linear parameter estimation. In the following, we test the proposed method on a widely-used benchmark problem and compare the results with other state-of-art algorithms. Finally, the proposed method is applied to two microarray datasets for missing value imputation. RBFNs are considered to be incorporated in the quasi-ARX models, and all the experiments are implemented on a personal computer with CPU of Intel Core2 Duo T9400(2.53GHz) and RAM of 3G by Matlab 7.6. Lib SVM toolbox version 2.91 [61] is applied for SVR implementation, and ν-SVR is used with default parameter setting.

4.5.1 Identification of a Mathematical System

System under Study

This is a nonlinear example with seven input variables, which could be described as [35]

$$y(t)=\frac{\exp(-y^2(t-2)y(t-1))}{1+u^2(t-3)+y^2(t-2)}+\frac{\exp(0.5(u^2(t-2)+y^2(t-3)))y(t-2)}{1+u^2(t-2)+y^2(t-1)}$$
$$+\frac{\sin(u(t-1)y(t-3))y(t-3)}{1+u^2(t-1)+y^2(t-3)}+\frac{\sin(u(t-1)y(t-2))y(t-4)}{1+u^2(t-2)+y^2(t-2)}+u(t-1)+e(t)$$

where $e(t) \in (0, 0.5)$ is white noise. The system is excited by 1000 random sequence with the amplitude between -1 and 1, and the first 300 data sample of $y(t)$ and $u(t)$ is illustrated in Fig.4.5. To test the obtained models by using the four identification approaches, a set of 800 input-output data is sampled as test data, and the input data is described as

$$u(t) = \begin{cases} \sin(2\pi t/250) & \text{if } t < 500 \\ 0.8\sin(2\pi t/250) + 0.2\sin(2\pi t/25) & \text{otherwise.} \end{cases}$$

Results and Analysis

In this simulation, the nonlinear parameters are determined by k-means clustering algorithm. The number of clusters is equivalent to the value of M, which determines the nonlinearity of the quasi-linear kernel mapping. We will test root mean square error (RMSE) of the simulated model under

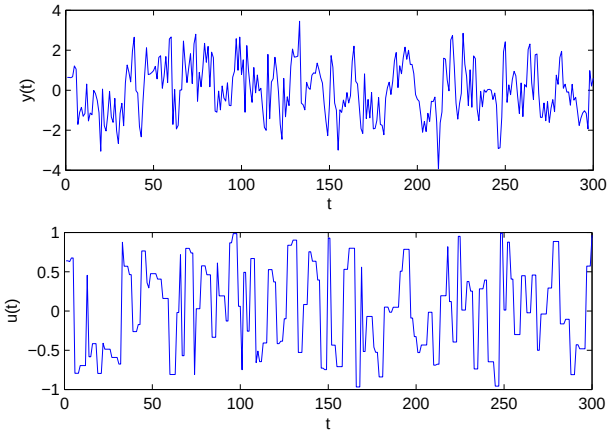


Figure 4.5: Training data of the mathematical system

different value of M, and the scale parameter λ in Eq.(4.3.5) is set to 0.05 in all the tests. Figure 4.6 shows the actual nonlinear parameter estimation results under the situation $M = 5, 6, 7, 8$, where the projections of constructed RBFs on the first dimension of input space are given.

Consider the randomness of k-means clustering results, simulations are implemented 50 times for each case, the simulation results are given in Tab.4.1. In this table, when M is as small as 1

Table 4.1: Simulation results under different value of M

M	RMSE(Mean±Std)	Min RMSE
1	0.340±0.000	0.340
2	0.184±0.000	0.184
3	0.176±0.023	0.155
4	0.205±0.004	0.201
5	0.222±0.032	0.183
6	0.221±0.034	0.170
7	0.210±0.036	0.143
8	0.214±0.039	0.137
9	0.208±0.032	0.149
10	0.207±0.034	0.152
20	0.271±0.045	0.199
30	0.323±0.029	0.243
40	0.336±0.023	0.257

or 2, the unique results are generated. Although both the mean and minimal value of RMSE have deterioration at the beginning of the M increasing, it is known from the tendency that performance

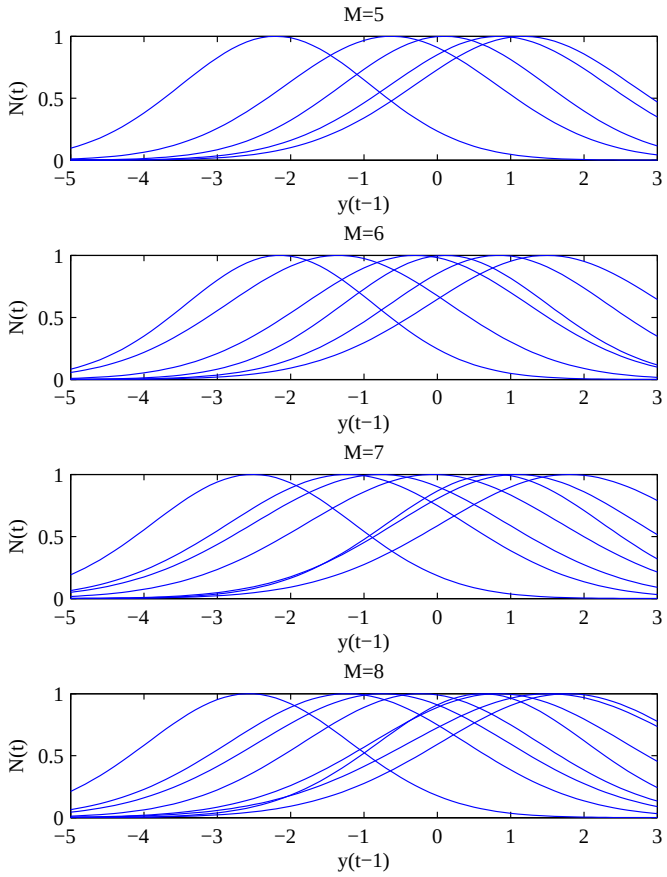
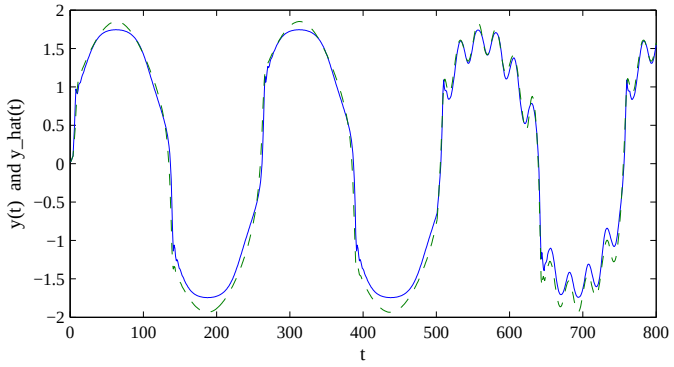
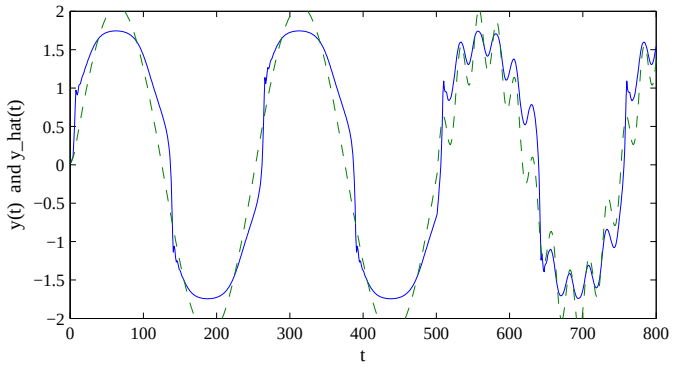


Figure 4.6: Results of constructed RBFs on the first dimension

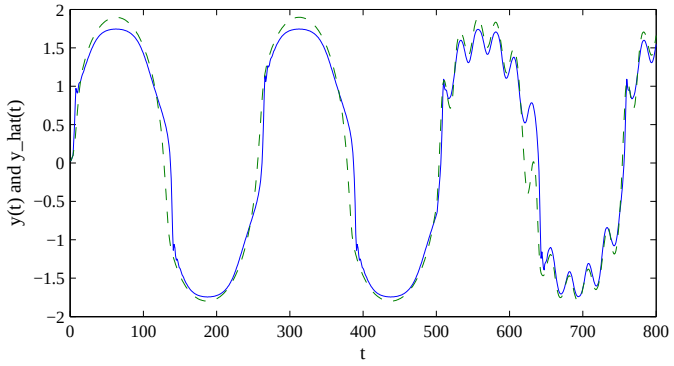
of the simulated model is improved firstly by increasing of the kernel mapping nonlinearity, then becomes worse with the continuous growing of the M. In other words, nonlinearity of the quasi-linear kernel mapping is tunable, and the appropriate nonlinear mapping can fit the problem well. In order to show the effectiveness of the quasi-linear kernel, it is compared with results from SVR with linear kernel and Gaussian kernel function as shown in Fig.4.7, where the solid line shows the system true output and the dashed line denotes the simulated model output. Figure 4.7(a) shows the result of a simulation with the proposed quasi-linear kernel on the test data, which gives a RMSE of 0.130 ($M = 8$). In contrast, the identified models by using SVR with linear kernel and Gaussian kernel are simulated in Fig.4.7(b) and Fig.4.7(c), which give RMSE of 0.345 and 0.258,



(a) Simulation result using the quasi-linear kernel



(b) Simulation result using linear kernel



(c) Simulation result using Gaussian kernel

Figure 4.7: Result Comparison of the mathematical system: (a) shows simulation result using the quasi-linear kernel; (b) shows simulation result using linear kernel; (c) shows simulation result using Gaussian kernel.

respectively. The parameter of gaussian kernel function is optimized as $\gamma = 0.05$. Therefore, SVR with the quasi-linear kernel is considered to own the appropriate nonlinearity and outperforms the one from Gaussian kernel and linear kernel.

4.5.2 Identification of a Contaminated System

System under Study

A benchmark mathematical problem is carried out to show the robustness of SVR for linear parameter estimation. It is taken from Narendra [40], which contains rather strong nonlinearity. The description is given as:

$$y(t) = f[y(t-1), y(t-2), y(t-3), u(t-1), u(t-2)] + e(t)$$

where

$$f[x_1, x_2, x_3, x_4, x_5] = \frac{x_1 x_2 x_3 x_5 (x_3 - 1) + x_4}{1 + x_2^2 + x_3^2}$$

where $e(t)$ is generated as stochastic impulse signals where the location of impulse is random and the amplitude is fixed. The first 100 noise samples with amplitude of -1 and +1 are illustrated in Fig.4.8. It is excited by 1000 random sequence with the amplitude between -1 and 1, and tested by

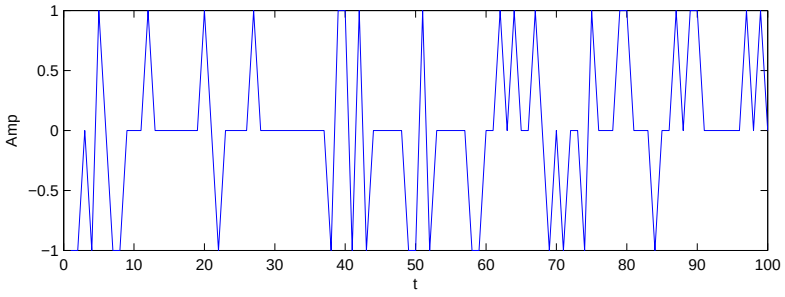


Figure 4.8: The first 100 samples of the impulse noise.

a set of 800 input-output data, which are the same with the previous example.

Results and analysis

Firstly, the noises with different degrees of amplitude are added on the system, and linear SVR and LS based approach are used to estimate linear parameters of the quasi-ARX model on test data. It should be mentioned that both the linear parameter estimation approaches are suffered from the

same disturbance and the influence from clustering partition method in the comparison. Affinity propagation (AP) algorithm [59] is applied for input space partition, which is recently introduced for exemplar-based clustering. The main advantage of this algorithm is that it can find good partitions of data and associate each partition with its most prototypical data point (exemplar) such that the similarity between points to their exemplar is maximized, and the overall cost associated with making a point an exemplar is minimized. Therefore, reasonable partitions can be generated automatically without predetermining the number of clusters, and the results are shown in Tab.4.2.

Table 4.2: Results of contaminated system identification with different amplitude of noise.

Amp	0	0.5	1	1.5	2
RMSE (LS)	0.041	0.061±0.020	0.077±0.027	0.095±0.038	0.110±0.043
RMSE (SVR)	0.030	0.051±0.014	0.054±0.015	0.057±0.015	0.060±0.019
Frequency		81	92	93	99

In this table, the term of Amp indicates the amplitudes of stochastic impulse noise added onto the training data, which are illustrated from 0 to 2. Due to the stochastic property of the noise, 100 times Monte Carols test is proceeded, and the statistical results of simulation RMSE over test data are given in the way of "mean value±standard deviation". What's more, the term of Frequency denotes how many times linear SVR outperforms LS based approach.

From Tab.4.2, it can be found that when the system is not contaminated by stochastic impulse noise, LS algorithm could perform a compatible accuracy with linear SVR. However, when the disturbance is added, linear SVR outperforms LS algorithm both in the mean values and the standard deviations of simulation RMSE. In other words, linear SVR based approach is less sensitive to the stochastic impulse noise than LS algorithm. In addition, it should be mentioned that, linear SVR outmatches LS algorithm according to the Frequency values from 100 tests, and the value almost increases with the amplitude of noise. In order to give a further insight into influences of noise on the linear parameter estimation methods, the stochastic impulse noise is added on the system with amplitude of 0.5, and the amplitude is increased 0.5 each iteration. The results of 50 iterations are shown in Fig.4.9.

In Fig.4.9, the simulation RMSE over test data are given by using both LS and linear SVR based linear parameter estimation approaches with continually added stochastic impulse noise. It is obvious that the simulation RMSE from linear SVR are less than the ones from LS method, and it is much more smooth with the noise added. As a consequence, it could be said that the linear SVR based estimation method is more robust when the system is contaminated by impulse noise.

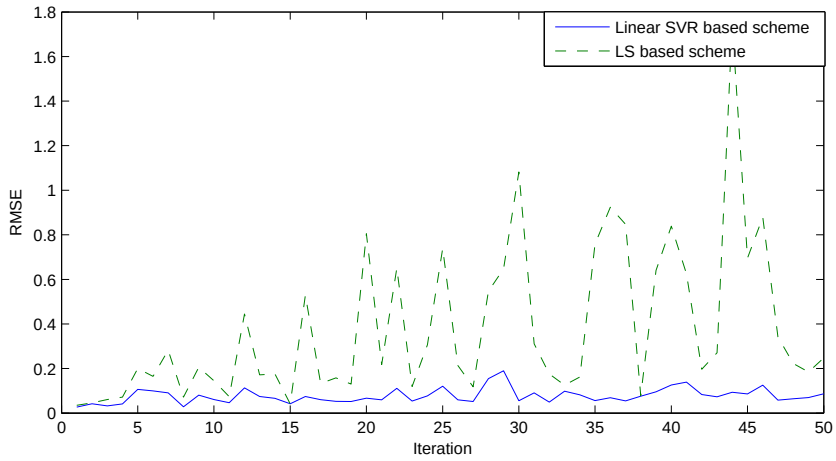


Figure 4.9: Results of influence of noise on the identification scheme.

However, the linear SVR performs only compatible with LS algorithm when white noise is added. In this example, when the system is contaminated by white noise satisfying $e \in (0, 0.5)$, there are only 54 times that linear SVR outperforms LS for linear parameter estimation in 100 times Monte Carlos test, and the mean value of RMSE from linear SVR and LS are 0.059 and 0.058, respectively. This situation is not improved when heavy white noise is further added on the system. When the white noise is set as $e \in (0, 1)$ and $e \in (0, 2)$ respectively, there are only 43 and 44 times that linear SVR based approach outperforms LS algorithm.

4.5.3 Identification of a Real System

System under Study

In order to further verify the effectiveness of the proposed method, a real system is tested which has been discussed by state-of-art algorithms as a benchmark problem. It is an example of modeling a hydraulic robot actuator, the position of a robot arm is controlled by a hydraulic actuator. The oil pressure in the actuator is controlled by the size of the valve opening through which the oil flows into the actuator. What we want to model is the dynamic relationship between the position of the valve $u(t)$ and the oil pressure $y(t)$, and the regressor is $y(t-1), y(t-2), y(t-3), u(t-1), u(t-2)$. A sample of 1024 pairs of $\{y(t), u(t)\}$ was observed as shown in Fig.4.10. The data was divided into two equal parts, the first 512 samples were used as training data, and the rest were used to test the trained network.

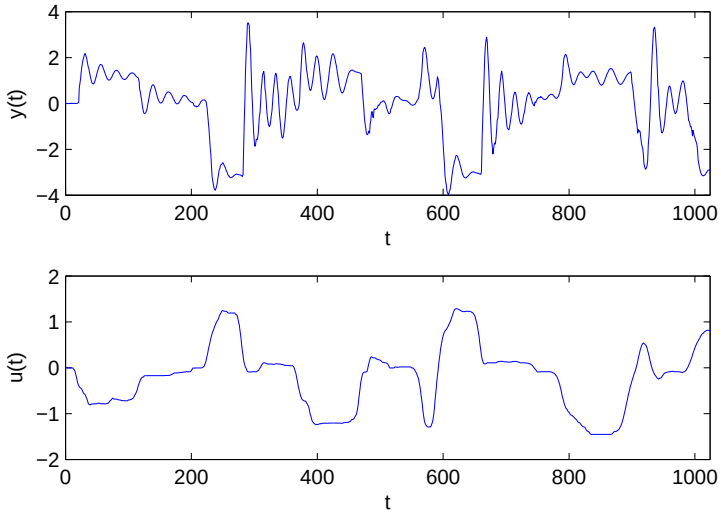


Figure 4.10: Measurement of system input and output.

Results and Analysis

Instead of k-means clustering algorithm, AP algorithm provides an automatical partition method for the nonlinear parameter determination and kernel learning in this example. 11 clusters are generated automatically, and the simulation of the model on the test data is shown in Fig.4.11, where the solid line represents the real measurements and the dashed line represents the results of simulation. It gives a RMS error of 0.507 when parameter $\lambda = 0.02$.

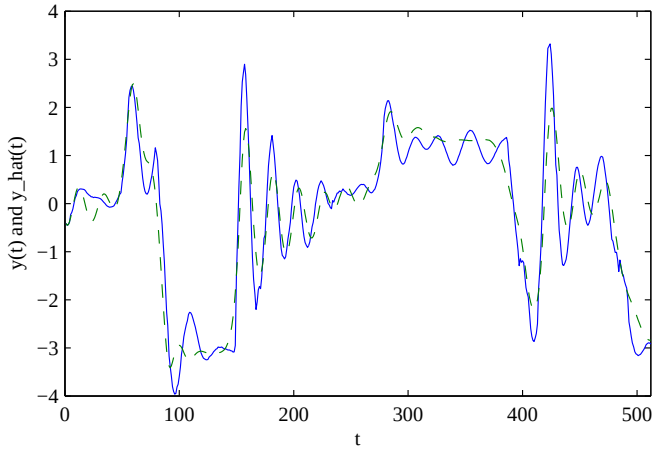


Figure 4.11: Simulation of SVR with the quasi-linear kernel on test data for the real system.

Table 4.3 shows the comparison results with NNs, WNs and SVR with linear (SVR/linear) and Gaussian (SVR/Gaussian) kernel. It is known that both the NNs and WNs own the universal

Table 4.3: Experiment results and comparison for the real system.

Models	RMSE
NNs	0.467
WNs	0.579
SVR/linear	1.008
SVR/Gaussian	0.280
SVR/quasi-linear	0.507

approximation ability, which give compatible results with our proposed model. Although SVR with Gaussian kernel in Ref.[72] is claimed to reduce the RMSE to 0.280 on the test data, it gives complex parameters setting which is considered case dependent.

4.5.4 Application to Microarray Missing Value Imputation

Problem Description

Microarray technology has been one of the most useful tools in functional genomics research [77]. It has been used widely in numerous studies over a broad range of biological disciplines, such as cancer classification, identification of genes relevant to a certain diagnosis or therapy, investigation the mechanism of drug action and cancer prognosis [78]. In a typical gene expression data matrix, the rows are the genes under investigation and the columns are the experimental conditions or time points. The gene expression data matrix is obtained by performing a series of microarray experiments on the same set of genes, one for each column. Unfortunately, datasets obtained from DNA microarray experiments often suffer from missing value problems. Diverse reasons lead to this problem, including insufficient resolution, image corruption, technical errors during hybridization, systematic errors on slides, or artifacts on the microarray [79, 80].

A number of algorithms have been proposed to solve this problem, which can be divided into global and local approaches. Algorithms of global approach perform missing value imputation based on global correlation information derived from the entire data matrix. They assume the existence of a global covariance structure among all genes or samples in the expression matrix. When this assumption is not appropriate, i.e. when the genes exhibit dominant local similarity structures, their imputation becomes less accurate. Well known imputation algorithms in this category include SVD imputation (SVDimpute) [81] and Bayesian principal component analysis (BPCA) [82]. In contrast, local approaches exploit only local similarity structure in the data set for missing value

imputation. Only a subset of genes that exhibits high correlation with the gene containing the missing values is used to compute the missing values in the gene. Some of the earliest and well-known imputation algorithms, such as, K nearest-neighbor imputation (KNNimpute) [81, 83], least square imputation (LSimpute) [84], local least square imputation (LLSimpute) [85, 86, 87], are among this category.

In this simulation we concentrate on the local approaches. Consider the fact that even if genes in one microarray take local similar structures, the linear combination of similar no-missing-value genes is not enough to capture the relation of the data, which is nonlinear in nature. However, the nonlinearity between the similar genes is not strong. Moreover, microarray data often comprises of huge size of genes with only a small number of observations, therefore the widely used nonlinear regression techniques are prone to overfitting [53]. The SVR based method with the quasi-linear kernel is applied to this problem, which aims to provide appropriate nonlinearity for the local imputation method.

Datasets

Two real datasets have been used in our experiments. The first one is from 784 cell-cycle-regulated genes, and 474 genes and 14 experiments (SP.CYCLE) are obtained after removing all gene rows that have missing values. It was designed to test how much an imputing method can take advantage of strongly correlated genes in estimating missing values. The second dataset is the cDNA microarray data relevant to human colorectal cancer (TA.CRC) and contains 758 genes in 50 samples. The two datasets have been used in studies of LLSimputation and BPCA imputation method.

Implementation

In order to generate missing values from the complete datasets, a randomly selected values are artificially missed under a given missing rate. In order to evaluate similarity between the genes, we give average values to the missing ones in each gene, then the imputation is implemented one by one. The flowchart of the proposed method for missing value imputation is given in Fig.4.12, where AP clustering algorithm is implemented, and only 2 clusters are generated automatically in both two datasets. Moreover, since missing values used in our experiments are introduced artificially, the performance of imputation methods are evaluated by comparing the estimated values with the corresponding real values. The metric to assess the accuracy of estimation is normalized root mean

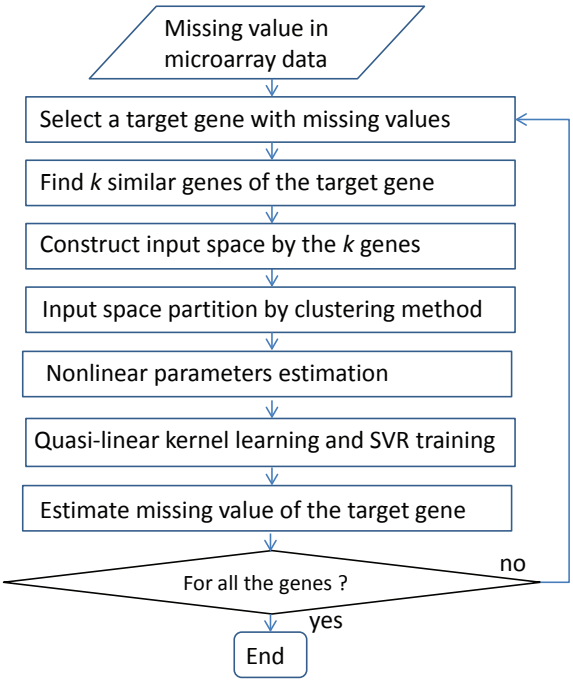


Figure 4.12: Flowchart of microarray missing value imputation

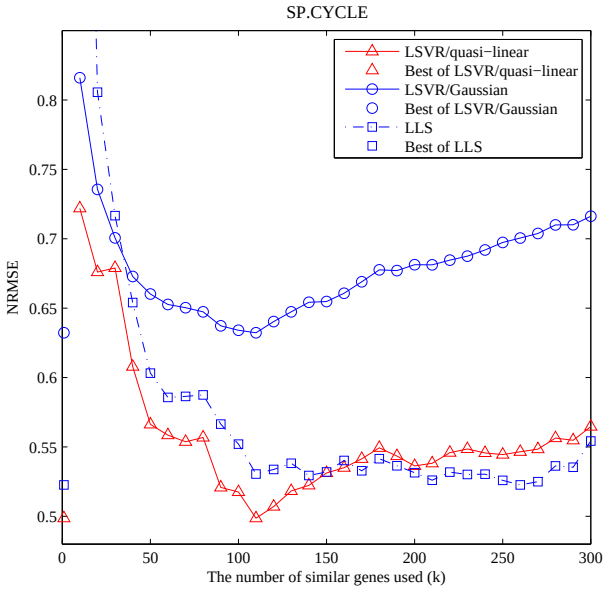
squared error (NRMSE):

$$\text{NRMSE} = \sqrt{\frac{\text{mean}[(y_{guess} - y_{ans})^2]}{\text{variance}[y_{ans}]}}$$

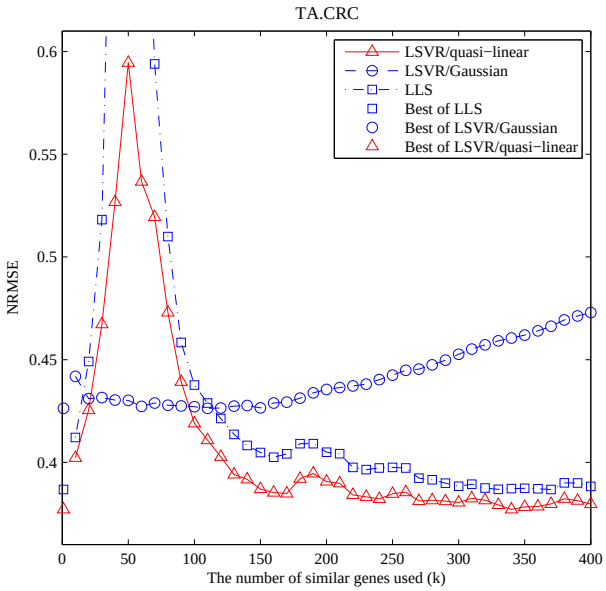
where y_{guess} and y_{ans} are estimated and real vectors for the artificial missing values.

Experimental Results

In Fig.4.13, the experimental results are shown for SP.CYCLE and TA.CRC dataset with missing data rate of 1%. In both cases, the NRMSE values for all the methods are tested with various values of k, and the best results of each method are shown on the y-axis. It is found that LLSimputation (LLS) method outperform local SVR with Gaussian kernel (LSVR/Gaussian), and the local SVR with the quasi-linear kernel based imputation approach shows the best performance among all the methods compared. What's more, as the percentage of missing value increased, the NRMSE values of both two dataset are also shown in Tab.4.4. It is found that local SVR with the quasi-linear kernel outperforms both linear and nonlinear local imputation methods.



(a)



(b)

Figure 4.13: Comparison of the NRMSE and effect of the k-value on SP.CYCLE and TA.CRC dataset with 1% entries of each dataset.

Table 4.4: NRMSE with missing percentage of 1%, 3%, 5%, 7% and 10%

	SP.CYCLE				
missing rate	1%	3%	5%	7%	10%
LLS	0.523	0.541	0.610	0.637	0.635
LSVR/Gaussian	0.632	0.581	0.641	0.686	0.677
LSVR/quasi-linear	**0.499**	**0.541**	**0.587**	**0.629**	**0.617**
	TA.CRC				
missing rate	1%	3%	5%	7%	10%
LLS	0.387	0.370	0.437	0.408	0.391
LSVR/Gaussian	0.426	0.427	0.446	0.431	0.408
LSVR/quasi-linear	**0.377**	**0.365**	**0.434**	**0.407**	**0.389**

4.6 Discussions and Conclusions

4.6.1 Discussions

The construction of the Mercer kernel functions with appropriate nonlinear mapping in a specific application are far from fully-understood. Exploring of new techniques and systematic methodology to learn a kernel function with physically meaningful kernel mapping is important for SVMs. The quasi-linear kernel is derived from the quasi-ARX models, whose nonlinear parameters are interpretable thus can be learnt by means of prior knowledge. It is found from the numerical simulations that nonlinearity of the quasi-linear kernel can be adjusted by the M, which is the number of clusters of input space data. There are two ways to determine this value in this chapter: one is to determine by trail-and-error as shown in Tab.4.1; another is by using automatic partition technique such as AP clustering algorithm. In fact, these methods try to determine nonlinearity of the quasi-linear kernel from the view of clustering partition, and grid partition methods can be used for kernel learning similarly.

What's more, RBFs are built on all the clustered sub-regions, and they can be explained as interpolations for a multi-local linear model. Therefore, the quasi-linear kernel as shown in Eq.(4.4.6) can be correspondingly explained as local linear kernel with interpolation. Similar to the multi-local linear model, the more local partitions it uses, the higher nonlinearity is the quasi-linear kernel.

4.6.2 Conclusions

In this chapter, a novel kernel method is proposed for the quasi-ARX model identification. SVR based identification approach is introduced to provide robust performance for the quasi-ARX model linear parameter estimation. Moreover, computational burden of grid partition method in Chapter 3

is reduced, since the SVR transforms the original linear parameter estimation problem into a dual form of quadratic optimization in the feature space, which is independent to the input dimension.

Different from the conventional kernel method, the proposed kernel owns explicit and physically meaningful kernel mapping, which is learnt from the quasi-ARX modeling with prior knowledge. Therefore, appropriate feature space can be reproduced, and nonlinearity of the kernel mapping is tunable between the linear and some existed nonlinear kernel functions. The proposed kernel function is thereby named quasi-linear kernel.

The effectiveness and usefulness are shown by simulation of numerical and real systems and application to microarray missing value imputation problem. SVR with the quasi-linear kernel outperforms the ones with linear kernel and Gaussian kernel in the numerical cases, and it generates compatible results with state-of-art algorithms in the real benchmark problem. What's more, the performance is improved by means of the quasi-linear kernel in microarray missing value imputation problem.

Chapter 5

Identification of Quasi-ARX NN Model for Polynomial Systems

5.1 Introduction

Recently, neural network (NN) has been reported as a global approach to describe the input-output relationship with universal approximation ability [6, 7]. Therefore, it is nature to hybrid the global property and outstanding fitting capability into the quasi-ARX models by incorporating NN into core-part of the model. In this way, the quasi-ARX NN (Q-ARX-NN) model is expected to provide not only potential approximation ability, but also useful structure for nonlinear controller design [35, 39].

On the other hand, polynomial NARX (Nonlinear AutoRegressive with eXogenous inputs) models, which are linear-in-parameter [88], have attracted much attention since the polynomial terms are often amenable to a direct physical interpretation, especially in some specific applications, such as mechanical modeling for vibration analysis, non-linear frequency analysis [89] and nonlinear control [90, 91]. However, it is known that polynomial NARX model identification remains a difficult task especially when the model is with high order, because a large pool of model terms has to be considered initially [92, 93], from which a useful model is then generated based on the parsimonious principle, of selecting the smallest possible model [1]. What's more, the number of candidate terms grows with increasing the order of the model and the maximum delay of the input and output signals drastically [32, 94]. Nevertheless, experience shows that provided the significant terms in polynomial NARX model can be detected, models with about 10 terms from the candidate pool are usually sufficient to capture the dynamics of highly nonlinear single-input and single-output (SISO) processes [18, 95, 96]. Most candidate terms are either redundant or make

very little contribution to the system output and can be removed from the model. Therefore, a key point of polynomial NARX model identification is how to select a moderate number of necessary monomial terms from a very huge number of candidates [97, 30, 32].

This chapter aims to identify the Q-ARX-NN model for polynomial systems. Since evolutionary optimization based approaches for polynomial system identification is efficient when the searching space is small, it is highly motivated to combine a optimization approach with a pre-screening process, in which a identified quasi-ARX NN model is used to reduce the searching space. Therefore, a two-step scheme is proposed, and in the first step, two importance indices are proposed to reduce the candidate term pool. Firstly the Q-ARX-NN model is used to approximate the system under study; the identified model is then transformed to polynomial form by performing a Taylor expansion; a reasonable number of important monomial terms is finally selected based on an pre-defined importance index, which is call Importance Index 1. Another importance index is introduced to further select terms with big contribution to the system output by error reduce ratio (ERR) from a simplified orthogonal least square (OLS) algorithm [29]. It is called Importance Index 2 and works as an auxiliary approach to Importance Index 1, which evaluates terms from a different perspective to reduce the original candidate pool, although both the two processes are not accurate. In the second step, multi-objective evolutionary algorithm (MOEA) is applied to search in a relative small space, which is aimed to optimize the root mean square error for one-step-ahead prediction over the training data (RMSE-T) and model complexity simultaneously [98]. In order to detect the best polynomial model from all the generated non-dominated solutions, the Pareto solution with the least root mean square error for one-step-ahead prediction on an independent validation data (RMSE-V) will be preferred. Although the similar idea of two-stage identification [19] has been reviewed before, it only focused on improving the compactness of the model obtained by the forward stepwise methods, rather than searching space reduction for optimization. In the experiments, two cases are simulated, and the results are compared with the ones generated by OLS [18], GA based method [99] and GP based method [29] as well.

This chapter is organized as follows: Section 5.2 gives a brief review of polynomial system identification methods. Section 5.3 introduces identification of the Q-ARX-NN model. Section 5.4 discusses the two-step identification scheme in detail. Section 5.5 provides numerical simulations to demonstrate the effectiveness of the proposed two-step identification method, and the result is also compared with other algorithms. Section 5.6 presents the discussions and conclusions.

5.2 A Brief Review of Polynomial System Identification

So far, there are mainly two kinds of approaches existed to determine polynomial NARX model structure [29]. The first one is selection based method, which generates all the possible monomial terms and select the best ones. Among so many methods of this category, orthogonal least squares (OLS) [18, 94, 31] is perhaps among the most popular, and the elegance of it lies in that, the net decrease in the cost function can be explicitly formulated as each new term is selected for inclusion in the model with the parameters obtained using backward substitution. However, if a large mount of monomial terms are included initially, it will become prohibitively expensive as its cost increases super linearly with the number of candidates [100]. What's more, it has been pointed out that it cannot guarantee that the resultant model is globally optimized [101]. Recently, a neural computation based polynomial NARX model identification has been proposed by performing a Taylor expansion of an identified neural network. However, it can only realize low accuracy identification and be applied to very simple case since the NN is easily trapped into a local minima [102].

In contrast, the second approach is based on optimization method, among which the original task is transferred into an optimization problem and solved using searching algorithms, such as Genetic Algorithm (GA) [100, 28, 27], Genetic Programming (GP) [29] and multi-objective optimization based methods [103, 104, 105, 106]. Although they can effectively search many local optima and thereby increase the possibility of finding the global optimum, cases with thousands of candidate terms are considered intractable because of the so big searching space. Ref. [99] claims that a hierarchical encoding technique is introduced to be effective for identifying polynomial models of relatively high-order, but as told by the authors, the process is very time-consuming and easily traps into a local optimum.

Consider the following SISO NARX system whose input-output relation described by:

$$y(t) = g(\varphi(t)) + e(t) \tag{5.2.1}$$

$$\begin{aligned} \varphi(t) &= [y(t-1), \cdots, y(t-n_y), u(t-1), \cdots, u(t-n_u)]^T \\ &= [x_1(t), \cdots, x_{n_y}(t), x_{n_y+1}(t), \cdots, x_n(t)]^T \end{aligned} \tag{5.2.2}$$

where $u(t) \in R$, $y(t) \in R$, $e(t) \in R$ are the system input, the system output and a stochastic noise of zero-mean at time t ($t = 1, 2, \cdots$), respectively. $g(\cdot) : R^{n=n_u+n_y} \rightarrow R$ is an unknown continuous function (black-box) describing the dynamics of system under study, and $\varphi(t) \in R^n$ is the regression vector composed of delayed input-output data. n_u and n_y are unknown maximum delays of the input and output respectively.

When identifying the system, we use the following nonlinear polynomial model with an order of q

$$\begin{aligned} y(t) &= y_0 \\ &+ \sum_{i_1=1}^{n} \alpha_{i_1} x_{i_1}(t) + \sum_{i_1=1}^{n} \sum_{i_2=i_1}^{n} \alpha_{i_1 i_2} x_{i_1}(t) x_{i_2}(t) + \cdots \\ &+ \sum_{i_1=1}^{n} \cdots \sum_{i_q=i_{q-1}}^{n} \alpha_{i_1 \cdots i_q} x_{i_1}(t) \cdots x_{i_q}(t) + e(t) \\ &= y_0 + y_1(t) + \cdots + y_i(t) + \cdots + y_M(t) + e(t) \qquad (5.2.3) \\ &= \Phi^T(t)\Theta + e(t) \qquad (5.2.4) \end{aligned}$$

where M is the maximum number of the candidate terms, $y_i(t)$ is the i-th monomial terms, the vectors Θ and $\Phi(t)$ are defined as follows:

$$\Theta = [y_0, \alpha_{i_1}, \alpha_{i_1 i_2}, \alpha_{i_1 \cdots i_q}, \cdots]^T$$

$$\Phi(t) = [1, x_{i_1}(t), x_{i_1}(t)x_{i_2}(t), \cdots, x_{i_1}(t) \cdots x_{i_q}(t), \cdots]^T$$
$$i_1, i_2, \cdots, i_q = 1, \cdots, n.$$

Although the polynomial model is linear-in-parameter, it consists of a huge number of candidate monomial terms. When the order q of polynomial model and the dimension n of the regression vector $\varphi(t)$ increase, the maximum number of the candidate terms increases drastically by

$$M = \sum_{i=0}^{q} n_i \qquad (5.2.5)$$

where

$$n_i = \frac{n_{i-1}(n_y + n_u + i - 1)}{i}, n_0 = 1. \qquad (5.2.6)$$

GA based method with a novel hierarchical encoding technique [99] has been proposed, and it is considered to be suitable to the structure of the nonlinear polynomial models. The problem is that long time is needed for identifying complex polynomial NARX models with relatively high-order. To improve the efficiency of GA, we propose a two-step identification scheme [107, 95] combining multi-objective searching with the pre-screening process.

5.3 Identification of Q-ARX-NN Model

5.3.1 Q-ARX-NN Model

The Q-ARX-NN model is a flexible nonlinear model with various linear properties [46, 35, 47]. Figure 5.1 shows structure of the Q-ARX-NN model. It can be described by

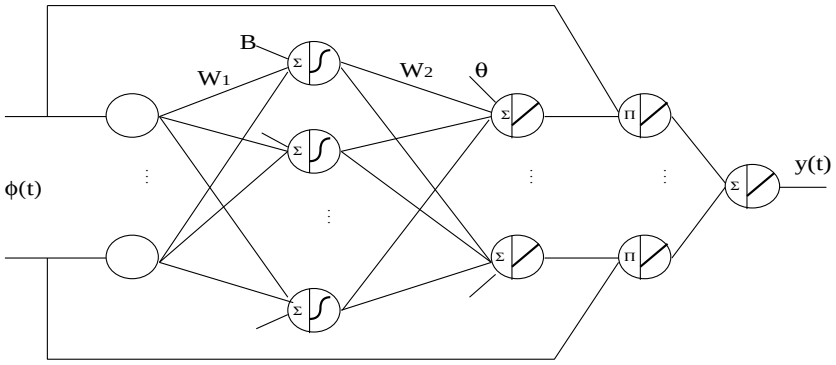


Figure 5.1: Structure of the Q-ARX-NN model.

$$y(t) = \phi^T(t)\theta + \phi^T(t)W_2\Gamma(W_1\phi(t) + B) + e(t) \tag{5.3.1}$$

where $\phi(t) = [y(t-1)\ \cdots\ y(t-m_y)\ u(t-1)\ \cdots\ u(t-m_u)]^T$,[1] $W_1 \in \mathcal{R}^{n_h \times m}$ $(m = m_u + m_y)$, $W_2 \in \mathcal{R}^{m \times n_h}$ are the weight matrices of the first and second layers, $B \in \mathcal{R}^{n_h \times 1}$ is the bias vector of hidden nodes, $\theta \in \mathcal{R}^{m \times 1}$ is the bias vector of output nodes, and Γ is the diagonal nonlinear operator with identical sigmoidal elements κ (i.e., $\kappa(\beta/\alpha) = \frac{1-e^{-\beta/\alpha}}{1+e^{-\beta/\alpha}}$) ($\alpha$ is an appropriate positive value), n_h is the number of nodes in hidden layer.

Compared to conventional feedforward NNs, the Q-ARX-NN model has easy-to-use structure and has been introduced efficient algorithms for the parameter estimation [47].

5.3.2 Hierarchical Parameter Estimation

This model contains two parts: the first term of Eq.(5.3.1) is a linear ARX model part, while the other is a nonlinear part. Therefore, the quasi-ARX model describes a linear approximation by means of θ, which is different with other parameters in the model. This feature allows us to use a dual loop learning algorithm for estimation. In Loop I, the parameter vectors W_1, W_2 and B are fixed and treated as constant vector, then θ can be estimated by

$$z_L(t) = \phi^T(t)\theta$$

where θ can be estimated by the well-known LS algorithm, and $z_L(t)$ is the output of linear ARX submodel calculated by

$$z_L(t) = y(t) - \phi^T(t)W_2\Gamma(W_1\phi(t) + B).$$

[1] m_u and m_y are roughly guessed values of n_u and n_y, which are usually larger than the exact values of n_u and n_y.

In Loop II, the parameter vector θ is fixed and treated as constant vector, then W_1, W_2 and B are estimated as parameters of nonlinear submodel

$$z_N(t) = \phi^T(t) W_2 \Gamma(W_1 \phi(t) + B)$$

where a backpropagation (BP) algorithm is applied to train the parameters, and $z_N(t)$ is the output of nonlinear submodel calculated by

$$z_N(t) = y(t) - \phi^T(t)\theta.$$

In this way, a hierarchical training scheme is described as follows.

Step1 set $\theta = 0$, and assign small initial values of W_1, W_2 and B;

Step2 calculate $z_L(t)$, then estimate θ by LS algorithm for the linear submodel.

Step3 calculate $z_N(t)$, then estimate W_1, W_2 and B for the nonlinear submodel. This is realized by BP algorithm, but the BP is only performed for a few epochs in order to avoid over-fitting;

Step4 stop if pre-specific conditions are met, otherwise go to Step2 and repeat the estimation of θ and W_1, W_2, B.

It is found that the dual-loop algorithm can solve over-fitting or local minimum problem well.

5.4 Two-Step Identification Scheme

It is found in numerical experiments that optimization based approaches are very efficient when the size of searching space is not too large. Considering this fact, a two-step identification scheme is proposed, and the flowchart is shown in Fig.5.2.

5.4.1 Selection Based Pre-screening Step

To select important terms with big contribution to the output vector from a huge number of candidates efficiently, Importance Index 1 and 2 are used to select terms in turn. For simplicity, two corresponding selection processes are named as Preliminary Selection and Further Selection.

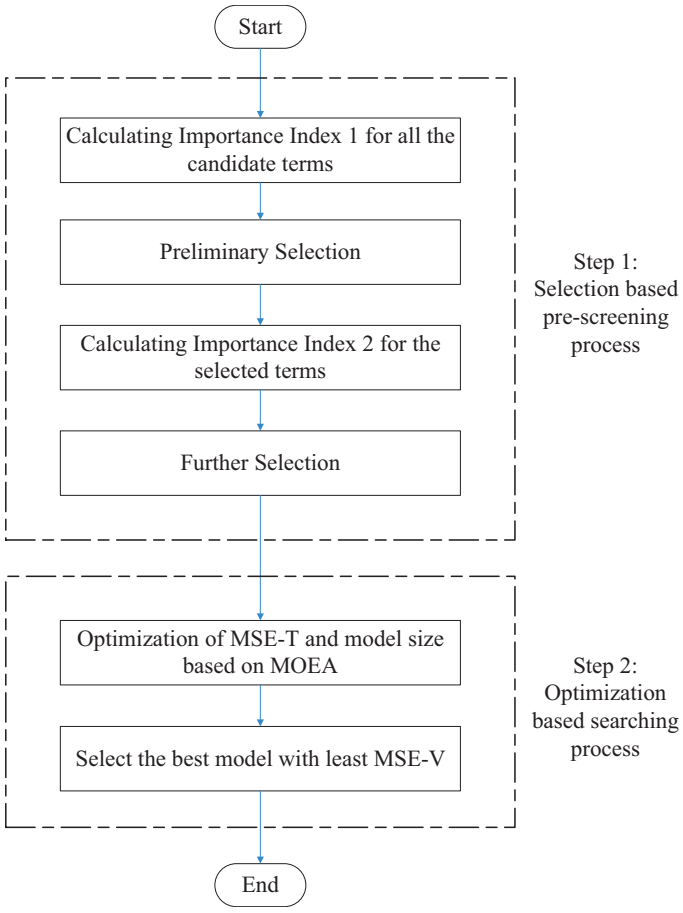


Figure 5.2: Flowchart of the two-step identification scheme.

Importance Index 1 for Preliminary Selection

In order to select important terms from a huge number of candidates, we introduce an importance index for the candidate terms, defined by

$$\mathcal{I}_i^{(1)} = \mathcal{I}(\frac{\text{var}[\mathbf{y}_i]}{\text{var}[\mathbf{y}]}) \tag{5.4.1}$$

where $\mathcal{I}(\cdot)$ is a pre-defined function based mainly on order of the monomial term, var$[\cdot]$ denotes variance, $\mathbf{y}_i$ is the i-th candidate term vector with coefficient and $\mathbf{y}$ is the output vector. To obtain the coefficients of each monomial term $\mathbf{y}_i$, the Q-ARX-NN model [107] is applied to identify the system firstly as Section 5.2 shown, then the model is transformed into a huge number of terms by

applying Taylor expansion to the sigmoid node function $\kappa(\frac{\beta}{\alpha})$ on a small region of $\frac{\beta}{\alpha} = 0$.

$$\begin{aligned} \kappa(\frac{\beta}{\alpha}) &= \sum_{l=0}^{l_{\max}} \frac{1}{l!}\kappa^{(l)}(0)(\frac{\beta}{\alpha})^l \\ &= \frac{1}{2\alpha}\beta - \frac{1}{24\alpha^3}\beta^3 + \frac{1}{240\alpha^5}\beta^5 - \cdots \end{aligned} \tag{5.4.2}$$

Using (5.4.2), Eq.(5.3.1) can be expanded to

$$\begin{aligned} y(t) &= \sum_{k=1}^{m}\theta_k x_k(t) + \frac{1}{2\alpha}\sum_{k=1}^{m} x_k(t)\sum_{j=1}^{n_h} w_{2kj}\left(\sum_{i=1}^{m} w_{1ji}x_i(t) + b_j\right) \\ &\quad -\frac{1}{24\alpha^3}\sum_{k=1}^{m} x_k(t)\sum_{j=1}^{n_h} w_{2kj}\left(\sum_{i=1}^{m} w_{1ji}x_i(t) + b_j\right)^3 \\ &\quad +\frac{1}{240\alpha^5}\sum_{k=1}^{m} x_k(t)\sum_{j=1}^{n_h} w_{2kj}\left(\sum_{i=1}^{m} w_{1ji}x_i(t) + b_j\right)^5 - \cdots + e(t) \\ &= y_1(t) + y_2(t) + \cdots + e(t) \end{aligned} \tag{5.4.3}$$

where $\beta = \sum_{i=1}^{m} w_{1ji}x_i(t) + b_j$, θ_k, b_j, w_{1ji} and w_{2kj} are the elements θ, B, W_1 and W_2, respectively, and $x_i(t)$ denotes the element of $\phi(t)$.

Importance Index 2 for Further Selection

OLS is considered as one of the most accurate and simplest method for polynomial model identification, and it uses the ERR to measure decrease in the variance of the output by a given term. Instead of the recursive manner of original OLS method, a simplified algorithm is introduced to estimate the ERR for all the selected important terms in one run, and it could highlight terms with larger contribution to output vector thus improve the accuracy of Preliminary Selection.

From the compact form of the polynomial NARX model represented as Eq.(5.2.4), in which Φ is the regression matrix, and Θ is the parameter vector, the OLS technique transforms the columns of the Φ matrix into a set of orthogonal basis vectors in order to inspect the individual contributions of each term. It is assumed that the regression matrix Φ can be orthogonally decomposed as $\Phi^T = WA$, where W is an matrix with orthogonal columns in the sense that $W^TW = D$ is a diagonal matrix, and A is an upper triangular matrix. Thus the sum of square of the output vector $\mathbf{y}$ is

$$< \mathbf{y}, \mathbf{y} >= \sum_{i=1}^{M} g_i^2 < \mathbf{w_i}, \mathbf{w_i} > + < \mathbf{e}, \mathbf{e} >, \tag{5.4.4}$$

and the ERR due to $\mathbf{w_i}$, which is the i-th column of W, is thus defined as the proportion of the output variance for all the terms

$$[\text{ERR}]_i = \frac{g_i^2 < \mathbf{w_i}, \mathbf{w_i} >}{< \mathbf{y}, \mathbf{y} >}, i = 1, \dots M, \tag{5.4.5}$$

where M is the size of candidate pool and $\mathbf{g}$ could be calculated in one run by

$$g_i = \frac{< \mathbf{w_i}, \mathbf{y} >}{< \mathbf{w_i}, \mathbf{w_i} >}. \tag{5.4.6}$$

Different from the recursive forward selection, there is no need to calculate ERR for each candidate when a new term is added, the regression matrix Φ is orthogonal decomposed directly, then ERR for all the candidate terms are calculated in one run. Obviously it will be processed fast, and provides a rough but simple way to evaluate the contribution of each monomial term to system output.

In both two Important Index, the principle of simplicity is applied, and the Importance Index 2 can be given by

$$\mathcal{I}_i^{(2)} = \frac{\text{ERR}_i}{e^{O_i}} \tag{5.4.7}$$

where O_i is the order of the ith term. Similar expression of $\mathcal{I}_i^{(1)}$ for Important Index 1 can also be obtained in the same way. Although Further Selection is not as accurate as OLS, as an auxiliary process of the Preliminary Selection, it could be used to prune candidate pool efficiently with all the necessary terms included.

5.4.2 MOEA Based Identification Step

In order to generate not only valid models of the systems but also parsimonious ones, two objectives are specified which addresses the themes of: 1) model performance and 2) model complexity. In fact, in most cases the model performance on training data will be improved with increasing of the model complexity, which may lead to overtraining. Therefore, RMSE-T and model complexity are expected to be minimized simultaneously, which are always conflict with each other, hence it's very natural to solve this problem by MOEA.

Coding for Polynomial NARX Model Identification

In the case of applying GA directly to the searching space containing a huge number of candidates, much effort and computing cost for encoding has to be done in order to improve the efficiency because of the large searching space [99, 28]. However, in our case the searching space containing

only the selected important terms is rather small, a simple real-valued encoding scheme may be used. Since each selected important monomial term has assigned with a number (No.), this number is used directly as gene of chromosome. For example, a chromosome with fixed length of 8, defined by

0	0	0	1	2	3	4	5

It represents a polynomial model consisting of monomial terms No.1, No.2, No.3, No.4 and No.5, where "0" denotes empty term.

Multi-objective Optimization Method

To make a compromise between model performance and generalization [32], there are two objectives to be minimized: RMSE-T and the model complexity. The two objectives could be represented as:

$$\text{obj1} = \sum_{t=1}^{N}[y(t) - \Phi^T(t)\hat{\Theta}]^2 \tag{5.4.8}$$

$$\text{obj2} = L \tag{5.4.9}$$

where $\hat{\Theta}$ is the least squares estimate of Θ, and L denotes model complexity, which corresponds to the number of monomial terms included in the model.

Non-dominated Sorting Genetic Algorithm-II (NSGA-II) [108] has been developed by K. Deb et al., which is a fast and elitist multi-objective evolutionary algorithm. In NSGA-II, we first create the offspring population Q_t (of size N) using the parent population P_t (of size N). The usual genetic operators such as single-point crossover and bit-wise mutation operators are used in this process. Next, we combine the two populations to form an intermediate population R_t of size $2N$. In the following, we evaluate the fitness of each offspring in the $2N$ population using the above objective functions. At this stage, we carry out non-dominated sorting procedure over the $2N$ population to rank and divide the individuals into different non-dominated fronts. Thereafter, we create the new parent population P_{t+1} by choosing individuals of the non-dominated fronts, one at a time. We choose the individuals of best ranked fronts first followed by the next-best and so on, till we obtain N individuals. Since the intermediate population R_t has a size of $2N$, we discard those fronts which could not be accommodated. In case there is space only for a part of a front in the new population, we use a crowded-distance operator to determine the individuals among those in the front that are from the least crowded regions. We choose such individuals so as to fill up the required number in the new population P_{t+1}.

Finally, the non-dominated solutions are generated which represent a set of possible optimal polynomial NARX models with different values of accuracy and model complexity, and the most parsimonious solution could be selected by using independent validation data. The one with the least RMSE-V would be considered as the model with both accuracy and generalization.

5.5 Numerical Simulations

To show efficiency of the proposed two-step scheme for polynomial system identification, two examples are tested, and they are the same with those of Ref. [99]. In fact, although the model structure of the first example is perfectly known, it is difficult to deal with it efficiently because the original candidate pool is very big, and the heavy white noise is added onto the training data. In the latter example, the model is highly nonlinear, and the perfect polynomial model structure does not exist. Even with state-of-art algorithms, this problem still becomes intractable because of the expensive computation. Both the experiments are implemented on a personal computer with CPU of Intel Core2 Duo T9400(2.53GHz) and RAM of 3G by Matlab 7.6. We compare our results with those of OLS method [18], GA based method [99] and GP based method [29]. In the proposed method, in order to make sure all the necessary monomial terms are included in the reduced searching space, a much bigger pool is often formed, and it could also exhibit the outstanding searching ability of MOEA compared with others. The specifications of MOEA are given in Tab.5.1.

Table 5.1: Parameter setting for EAs

	Example 1	Example 2
Maximum generation	250	600
Population	50	300
Chromosome length	10	
Tournament size	2	
Crossover possibility	0.7	
Mutation possibility	0.3	

5.5.1 Training Data

As the training data, 500 and 1000 input-output data sets are sampled from the examples when the systems are excited using random input sequences. For instance, Fig.5.3 shows the first 300 set of training data for Example 2.

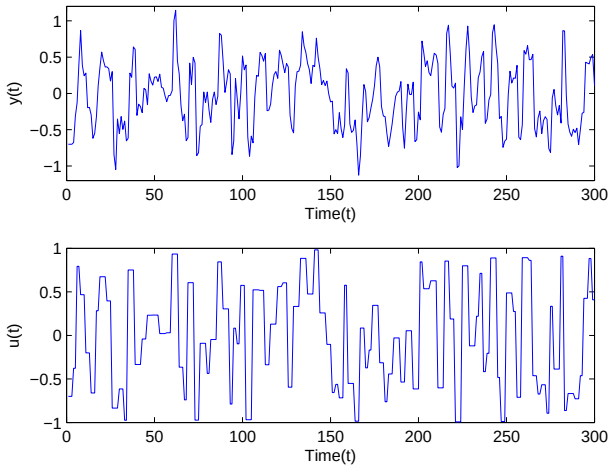


Figure 5.3: Training data samples for the rational system.

5.5.2 Identification of a Polynomial System

System under Study

Example 1: The system is governed by a polynomial model, described by

$$\begin{aligned} y(t) \;=\; & -0.5y(t-2) + 0.7u(t-1)y(t-1) + 0.6u^2(t-2) \\ & +\; 0.2y^3(t-1) - 0.7u^2(t-2)y(t-2) + e(t) \end{aligned}$$

where $e(t) \in (0, 0.2)$ is a white Gaussian noise with amplitude between -1.2 and 1.2.

Assumed the maximum input-output delay for identification is $n_u = 4$, $n_y = 4$, and maximum order is 5, thus the number of the candidate terms becomes 1286.

Identification Results

In the pre-screening step, 300 important terms are firstly determined by Preliminary Selection, then Further Selection is implemented according to the sorting of Importance Index 2, thus the candidate pool could be reduced to 200. In the second step, NSGA-II is used to identify the model structure from the reduced candidate pool. In order to decide the model size from a set of non-dominated solutions, new 200 validation data set are sampled in both two cases, in which magnitude of input data is between -1.2 and +1.2. The one with least RMSE-V is selected and considered as the best

model structure. The result of identified polynomial model is achieved as

$$\begin{aligned}\hat{y}(t) \;=\; & -0.499\hat{y}(t-2) + 0.649\hat{y}(t-1)u(t-1) + 0.692u^2(t-2) \\ & +\; +0.191\hat{y}^3(t-1) - 0.655\hat{y}(t-2)u(t-2)^2.\end{aligned}$$

Results of Pre-screening Step

In order to show the effectiveness of the pre-screening step, Tab.5.2 illustrates image of the table formed with ranks of true model terms, in which $Initial$ represents the original sequence number of terms when they are generated according to their orders initially. $Rank1$ denotes the term's rank value based on sorting of Importance Index 1, and $Rank2$ is Importance Index 2 based sorting results after Preliminary Selection.

Table 5.2: Ranks of true model terms in the pre-screening step for the polynomial system

True Model Terms	Initial	$Rank1$	$Rank2$	Order
$y(t-2)$	2	1	2	1
$u^2(t-2)$	39	3	3	2
$u(t-1)y(t-1)$	13	8	4	2
$y(t-2)u^2(t-2)$	**103**	47	20	3
$y^3(t-1)$	45	**86**	**32**	3

We can know from Tab.5.2 that all the true model terms become more significant after the pre-screening process. The ranking value of each term is finally improved, and the minimal candidate pool is also reduced from 103 to only 32 to contain all the necessary terms. What's more, in the pre-screening step, Preliminary Selection is improved by Importance Index 2 effectively, which reduced the minimal size of candidate pool from 86 to 32 for inclusion of the true model structure.

Results Comparison

In the first example, four approaches are compared.
Method 1: OLS method [18].
Method 2: GA based method [99].
Method 3: GP based method [29].
Method 4: MOEA based two-step identification method.

Because of the stochastic nature of evolutionary algorithms, Monte Carlo test will be proceeded with 100 times continuous run except Method 2 because it is time-consuming, and all the methods

are set with the same population size and generation. The result is shown in Tab.5.3.

Table 5.3: Experiment results and comparison for the polynomial system

Algorithm	Model structure	Monte Carlo Test
Method 1	Wrong	\
Method 2	Correct	\
Method 3	Correct	21/100
Method 4	Correct	93/100

As Tab.5.3 shows, OLS method could not find the right model structure, which is processed as Ref. [18] did and the first five best terms are chosen. Compare with the proposed method, other algorithms could find the correct model structure as well, but it should be mentioned that the GA based method (Method 2) costs much more time than others. Although the GP based identification method could identify the model quickly, it has only 21 percent possibility to find the correct solution. In contrast, MOEA based two-step identification scheme could make the correct identification as many as 93 times in 100 independent runs, and it is considered as the best approach to solve this problem.

5.5.3 Identification of a Rational System

System under Study

Example 2: The system is a nonlinear rational model studied by Narendra in 1990

$$y(t) = f[y(t-1), y(t-2), y(t-3), u(t-1), u(t-2)]$$

where

$$f[x_1, x_2, x_3, x_4, x_5] = \frac{x_1 x_2 x_3 x_5 (x_3 - 1) + x_4}{1 + x_2^2 + x_3^2}$$

where a white noise of $e(t) \in (0, 0.1)$ is further added.

It has been believed that this is a very complex and strongly nonlinear system, and there is not any perfect model structure existed. Assumed the maximum input-output delay for approximating is $n_u = 3$, $n_y = 4$, and maximum order of polynomial model is 8, so the maximum number of the candidate terms is 6434. Due to such the complicate model structure and huge candidate pool, few methods could be considered to identify the nonlinear polynomial model directly.

Identification Results

In the pre-screening step, 800 terms are firstly selected from 6434 candidate terms according to Importance Index 1, then Further Selection is implemented to reduce the candidate pool into 500 with little time cost. In the second step, NSGA-II is used to identify the model structure from the reduced candidate pool. In order to decide the model size from a set of non-dominated solutions, new 400 validation data set are sampled as the training data does, in which magnitude of input data is between -1.2 and +1.2. The one with least RMSE-V is selected and considered as the best model structure. The result of our proposed identification method is shown as

$$\begin{aligned} y(t) &= 0.912u(t-1) - 0.307y^2(t-3)u(t-1) - 0.416y(t-2)u(t-1)u(t-3) \\ &- 0.459y(t-1)y(t-2)y(t-3)u(t-2) + 0.343y(t-1)y^2(t-3)u^2(t-3). \end{aligned}$$

Result of Pre-screening Step

To explicit the effectiveness of the pre-screening step, the rank of identified model terms are listed in Tab. 5.4, in which the mean of $Initial$, $Rank1$, and $Rank2$ are same with the ones used in Example 1.

Table 5.4: Ranks of identified model terms in the pre-screening step for the rational system

Monomial Terms	Initial	$Rank1$	$Rank2$	Order
$u(t-1)$	5	1	1	1
$y^2(t-3)u(t-1)$	87	20	10	3
$y(t-2)u(t-1)u(t-3)$	81	12	15	3
$y(t-1)y(t-2)y(t-3)u(t-2)$	157	239	235	4
$y(t-1)y^2(t-3)u^2(t-3)$	**484**	**281**	**266**	5

It is found that the rank of most of the identified model terms are improved by the selection based pre-screening step. As a result, the minimal candidate pool size could be reduced from 484 to 281 with all necessary terms included in Preliminary Selection, and Further Selection makes the pool size threshold decreased to 266 with the help of the Importance Index 2 based sorting.

Results Comparison

Here are four different approaches used to be compared:
Method 1: OLS method with the pre-screening step.
Method 2: GA based method.

Method 3: GP based method.
Method 4: MOEA based two-step identification method.

We used OLS method with the pre-screening process in Method 1 because the original candidate pool is too large to deal with. The results generated from the different methods are shown in Tab.5.5.

Table 5.5: Experiment results for the rational system

Algorithm	Model structure		
Method 1	$u(t-1)$	$y^2(t-3)u(t-1)$	$y(t-2)u(t-1)u(t-3)$
	$y(t-1)y(t-3)u(t-2)u(t-3)$	$y(t-1)y(t-2)u^3(t-2)u^2(t-3)$	
Method 2	$u(t-1)$	$y^2(t-3)u(t-1)$	$y(t-2)u(t-1)u(t-3)$
	$y(t-2)u^2(t-2)$	$y(t-1)y(t-2)y(t-3)u(t-2)$	
Method 3	$y(t-2)$	$u^2(t-2)$	$u^2(t-2)y(t-2)$
	$y(t-1)u(t-1)$		
Method 4	$u(t-1)$	$y^2(t-3)u(t-1)$	$y(t-2)u(t-1)u(t-3)$
	$y(t-1)y(t-2)y(t-3)u(t-2)$	$y(t-1)y(t-3)^2u^2(t-3)$	

To test the obtained polynomial models, a 800 input-output data is sampled as test data, and the input data is described as

$$u(t) = \begin{cases} \sin(2\pi t/250) & \text{if } t \leq 500 \\ 0.8\sin(2\pi t/250) + 0.2\sin(2\pi t/25) & \text{otherwise.} \end{cases} \tag{5.5.1}$$

And the simulation of the obtained polynomial model from our proposed method is shown in Fig.5.4. The solid line represents the system true output, and the dash line denotes the polynomial model output. We find that the polynomial model obtained from the MOEA based two-step identification method represents the system quite well.

According to the generated models, the identification results from four methods are compared by RMSE-T and full prediction RMSE of test data (RMSE-Test). From Tab.5.6 we know that GP based method fails to simulate the polynomial model for this complex nonlinear system. Compare with the methods with selection based pre-screening process, GA based method works in low efficiency, and the author of this method told us that more than 50 hours will be needed for this processing. Moreover, although OLS method with the pre-screening process generates a similar simulated polynomial model with the proposed one, the model order is as high as 7, which is considered not easy to use from the view of application. In contrast, the MOEA based two-step identification method could generate nonlinear polynomial models with good performance.

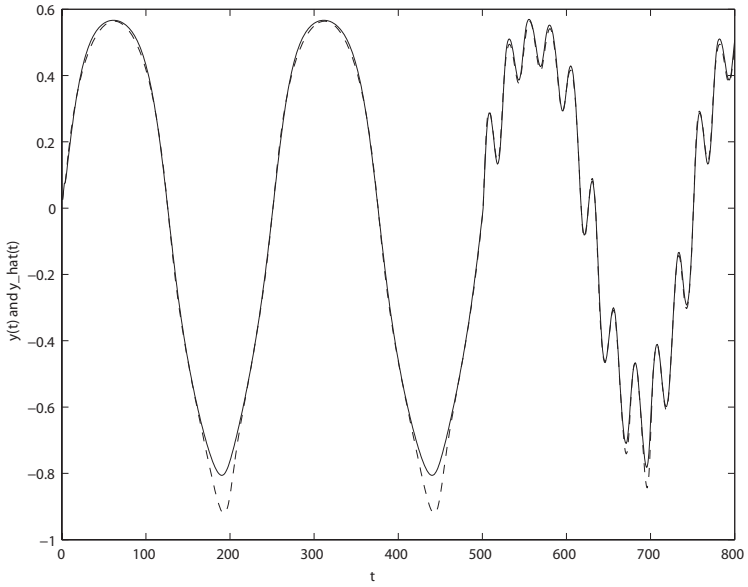


Figure 5.4: Simulation of obtained polynomial models on the test data.

Table 5.6: Results comparison for the rational system

Algorithm	RMSE-T	RMSE-Test	Model Size	Order
Method 1	0.120	0.026	5	7
Method 2	0.122	0.033	5	4
Method 3	0.483	Inf	4	3
Method 4	0.121	0.026	5	5

5.6 Discussions and Conclusions

5.6.1 Discussions

Two examples with large candidate pools are tested in simulations to illustrate the effectiveness of the proposed identification method. All the necessary terms become more important in the pre-screening step, although each of the importance index is not accurate, and there are some degradations in Preliminary Selection or Further Selection on its own. Especially, there is no perfect polynomial model structure existed in Example 2, and all the solutions are just approximations of the real system. We find that high-order terms' contributions to system output are small and similar with each other in this case, thus these terms become hard to be discriminated and sensitive to the

system noise. However, compared with the original sequence number of this term, the final ranking result after the pre-screening step is improved. In the future research, the mutual compensate mechanism for both the two importance indices would like to be studied, which aims to improve the accuracy and efficiency of the pre-screening process.

What's more, this chapter is mainly on the study and analysis of the two-step identification scheme for polynomial NARX models, and there is little effort endowed in the improvement of MOEA and the validation mechanism. However, it is enough to show the advantage of using MOEA to generate model structure, and effectiveness of the two-step scheme for polynomial NARX model identification is also presented. A wider comparison of the identification results from state-of-art algorithms will be given in the future.

5.6.2 Conclusions

In this chapter, the contribution is to introduce a two-step identification scheme for complex polynomial NARX model with high-order. In the first step, the pre-screening process is carried out based on two kinds of the importance indices, in which, the first one is based on the Q-ARX-NN model; and the second one is based on a simplified OLS algorithm. In the second step, MOEA is applied to determine a set of significant terms to be included in the polynomial model from the selected important candidate terms. Then validation data is applied to decide the best polynomial model from all the generated Pareto solutions.

Chapter 6

Conclusions

6.1 Summary

In this final chapter, we will give a summary for the whole thesis.

The quasi-ARX model provides an ARX-like linear structure with state-dependent coefficients, which can be parameterized by using nonlinear functions of past inputs and outputs. Due to the useful model structure and linear properties, the quasi-ARX models have been successfully applied in real applications, such as fault detection and nonlinear adaptive control. The main work of this thesis is to investigate effective nonlinear system identification methods using the quasi-ARX models.

The Chapter 2 and 3 concentrate on identification of the quasi-ARX models by introducing prior knowledge, where interpretable parameters are determined and fixed heuristically. In this way, the model flexibility is restricted, and the obtained models are also easily used in real applications. In Chapter 2, the quasi-ARX model is identified using clustering partition method. The wavelet network (WN) is utilized and incorporated in the quasi-ARX model, which provides an interpretable way for nonlinear parameter determination. The work of this chapter shows that

- Clustering partition method provides an heuristical guide for interpretable parameter estimation in the quasi-ARX models.
- From the aspect of application, a nonlinear adaptive controller is designed by using the quasi-ARX WN predictor, where linear parameters of the model can be adjusted only to cope with sudden changes on the system.

In Chapter 3, the quasi-ARX model is identified by incorporating neurofuzzy network (NFN) into the core-part. It can be considered as a simple and typical grid partition approach to introduce

prior knowledge for the quasi-ARX model. The fuzzy membership partition is implemented on each input variable, hence the total number of fuzzy rules generated increases with dimension of input space exponentially.

Heuristically, the system inputs linear to output are considered unnecessary for the core-part of the quasi-ARX model since the linear expression is sufficient to capture system dynamics. In this way, a parsimonious model structure is constructed by genetic algorithm (GA) based fuzzy rule selection scheme, where only significant input variables are selected for the incorporated NFN, hence the generalization is also improved.

In Chapter 4, the quasi-ARX model is identified using kernel learning approach, where the identification is implemented by means of an support vector regression (SVR) with quasi-linear kernel. The proposed kernel mapping is explicit and learnt from the quasi-ARX modeling with prior knowledge. The work of this chapter shows that

- SVR based linear parameter estimation not only introduces robust performance, but also alleviates computational cost for curse-of-dimensionality of grid partition based quasi-ARX model.

- The gap between existed linear and nonlinear kernel functions can be filled by the proposed quasi-linear kernel, in which prior knowledge is utilized efficiently.

- The quasi-linear kernel mapping can generate an appropriate and physically meaningful feature space for problems at hand, therefore the proposed identification method outperforms both the linear and nonlinear approaches in some real-world problems.

In Chapter 5, the quasi-ARX neural network model is identified for polynomial systems, where neural network (NN) is incorporated in the quasi-ARX model to provides an index to reduce the monomial candidate terms pool. A two-step identification scheme is proposed to cope with the huge candidate term pool efficiently.

- Firstly, a pre-screening process is carried out. The quasi-ARX NN model is identified initially, and transformed to polynomial form by performing a Taylor expansion; the variance of each term with the estimated coefficients are used for important term selection. A simplified orthogonal least square (OLS) algorithm is implemented as an auxiliary approach to reduce the candidate pool further.

- Secondly, multi-objective evolutionary algorithm (MOEA) is applied to determine a set of

significant terms to be included in the polynomial model from the selected important candidate terms.

6.2 Topics for Future Research

This thesis has achieved its goal and it also motivates several related topics for future research.

In the quasi-linear kernel learning, the appropriate nonlinearity (complexity) of the explicit kernel mapping plays the key role in identification. In this thesis, although some heuristic methods has been applied to show the effectiveness of the learnt composite kernel, systematic methodology is needed to constrain the complexity of the proposed kernel. Vapnik-Chervomenkis (VC) dimension theory provides a useful criterion for kernel complexity selection, which gives a guide from the perspective of upper bound of minimal VC dimension. Such research is currently in progress.

In polynomial system identification, we have only shown it is possible to select model structure efficiently by means of the two-step identification scheme. The research on MOEA has been a hot topic in recent years, which is important to the final result of identification. To improve the generalization and robustness of the identified models, it is worthy of further study in MOEA for system identification.

In this thesis, the quasi-ARX model is studied from the perspective of system identification. It has been pointed that the usefulness of the model structure is remarkable, especially in nonlinear control. It is believed that the quasi-ARX models with the proposed identification methods are promising to cope with complex and high-noised problems, and our future work may be directed in regulation and control in biological processes.

Appendix A

Orthogonal Least Square Method for Nonlinear System Identification

Term and variable selection are to pre-select a subset of significant variables or terms to form a parsimonious model structure, which can be obtained without loss of representational accuracy. In this appendix, the famous orthogonal least square (OLS) and error reduction ratio (ERR) approach are reviewed to detect the potential model structure for nonlinear system identification.

A.1 Introduction

Several types of model structures are available to approximate the unknown mapping of nonlinear system including parametric, and non-parametric models. It has been proved that most nonlinear dynamic systems can be described using the NARX (Non-linear auto regressive with exogenous inputs) model

$$\begin{aligned} y(t) &= g(\varphi(t))+e(t), \quad t=1,2,\cdots,N \\ \varphi(t) &= [y(t-1),\cdots,y(t-n_y),u(t-1),\cdots,u(t-n_u)]^T \\ &= [x_1(t),\cdots,x_{n_y}(t),x_{n_y+1}(t),\cdots,x_n(t)]^T \end{aligned}$$

where $u(t) \in R$, $y(t) \in R$, $e(t) \in R$ are the system input, the system output and a stochastic noise of zero-mean at time t, respectively. n_y and n_u are the maximum lags in the output and input, and $g(\cdot)$ is the unknown nonlinear mapping. The objective of system identification is to find a suitable model to approximate the underlying relationship using a set of input and output observations. When a polynomial expansion is selected the model becomes linear-in-parameter. Providing the model structure, or which terms are contained in the model has been determined, only the values of

parameter are unknown and the identification turns to a simple least square problem. Unfortunately, the model structure detection is not a easy task and rarely known as a priori. An inappropriate model structure would degrade the model performance badly no matter which algorithm is used.

One problem with the identification of linear-in-parameter models is that a very large pool of model terms has to be considered initially, from which a useful model is then generated based on the parsimonious principle, of selecting the smallest possible model. An efficient model structure determination approach has been developed based on OLS algorithm and the ERR criterion (an index indicating the significance of each model term), which was originally introduced to determine which terms should be included in a model [18]. This approach has been extensively studied and widely applied in non-linear system identification [30, 31, 63, 97, 109]. The forward OLS algorithm and the ERR approach for term detection will be briefly described below.

A.2 Orthogonal Least Square Method

Let's consider the generic form of the linear-in-parameter model as

$$y(t) = \sum_{i=0}^{M} \theta_i f_i(\varphi(t)) + e(t) \tag{A.2.1}$$

where $f_i(\cdot)$ are model terms which are formed by combining some of the input variables $x_i(t),\ i = 1, 2, \cdots, n$, where $f_0 \equiv 1$ corresponds to a constant term. M is the number of all the distinct terms, and θ_i are the unknown parameters to be estimated. In this way, a compact matrix form corresponding to (A.2.1) is

$$y(t) = \Phi^T(t)\Theta + e(t) \tag{A.2.2}$$

where

$$\begin{aligned} \Phi(t) &= [f_0(\varphi(t)), f_1(\varphi(t)), \cdots, f_M(\varphi(t))]^T \\ \Theta &= [\theta_0, \theta_1, \cdots, \theta_M]^T. \end{aligned}$$

Assume that the regression matrix Φ^T is full rank in columns and can be orthogonally decomposed as

$$\Phi^T = WA \tag{A.2.3}$$

where A is a $M \times M$ unit upper triangular matrix and W is a $N \times M$ matrix with orthogonal columns $w_1, w_2, \cdots, w_M$ in the sense that $W^TW = D$, in which D is a positive diagonal matrix.

The space spanned by the orthogonal basis $w_1, w_2, \cdots, w_M$ is the same as that spanned by the basis set $x_1, x_2, \ldots, x_M$ from Φ, and Eq.(A.2.2) can be expressed as

$$Y = (\Phi^T A^{-1})(A\Theta) + E = WG + E$$

where Y and E are the matrix form of $y(t)$ and $e(t)$, $G = [g_1, g_2, \cdots, g_M]^T$ is an auxiliary parameter vector, which can be calculated directly from Y and W by means of the property of orthogonality as

$$G = D^{-1}W^T Y$$

or

$$g_i = \frac{\langle Y, w_i \rangle}{\langle w_i, w_i \rangle}$$

where $\langle \cdot, \cdot \rangle$ denotes inner product. The parameter vector Θ is related by the equation $A\Theta = G$ and this can be solved using either a classical or modified Gram-Schmidt algorithm.

The number M of all the candidate terms in model (A.2.1) is often very large. Some of these terms may be redundant and should be removed to give a parsimonious model with only M_0 terms ($M_0 \ll M$). Detection of the significant model terms can be achieved using the OLS procedures described below.

Assume that the residual signal $e(t)$ in the model (A.2.1) is uncorrelated with the past outputs of the system, then the output variance can be expressed as

$$\frac{1}{N} Y^T Y = \frac{1}{N} \sum_{i=1}^{M} g_i^2 w_i^T w_i + \frac{1}{N} E^T E. \tag{A.2.4}$$

Note that the output variance consists of two parts, one is the desired output, $\frac{1}{N}\sum_{i=1}^{M} g_i^2 w_i^T w_i$, which can be explained by the regressors, and the other part, represents the unexplained variance. Thus the term $\frac{1}{N}\sum_{i=1}^{M} g_i^2 w_i^T w_i$ is the increment to the explained desired output variance brought by w_i, and the i-th error reduction ratio, ERR_i, introduced by $w_i, (i = 1, 2, \cdots, M)$, can be defined as

$$\text{ERR}_i = \frac{g_i^2 \langle w_i, w_i \rangle}{\langle Y, Y \rangle} \times 100\% = \frac{\langle Y, w_i \rangle^2}{\langle Y, Y \rangle \langle w_i, w_i \rangle} \times 100\%. \tag{A.2.5}$$

This ratio provides a simple but effective means for seeking a subset of significant regressors. The significant terms can be selected in a forward-regression manner according to the value of ERR_i. Several orthogonalization procedures, such as Gram-Schmidt, modified Gram-Schmidt and Householder transformation can be applied to implement the orthogonal decomposition as shown in reference [18]. The classic Gram-Schmidt method is shown as following.

The procedure computes A in (A.2.3) one column at a time and orthogonalizes Φ iteratively. It is assumed Φ^T is composed of M columns $p_1, p_2, \cdots, p_M$, at the k-th stage make the k-th column orthogonal to $k-1$ previously orthogonalized columns and repeat the operations for $k = 2, \cdots, M$. The computational procedure is represented as

$$\left.\begin{array}{l} w_1 = p_1 \\ \alpha_{ik} = \frac{\langle w_i, p_k \rangle}{\langle w_i, w_i \rangle}, \quad 1 \leqslant i < k \\ w_k = p_k - \sum_{i=1}^{k-1} \alpha_{ik} w_i \end{array}\right\} k = 2, \cdots, M$$

The selection procedure is continued until the stage M_0 stage when

$$1 - \sum_{i=1}^{M_0} \text{ERR}_i < \rho$$

where ρ ($0 < \rho \leqslant 1$) is a desired tolerance. It is seen that $1 - \sum_{i=1}^{M_0} \text{ERR}_i$ is the proportion of the unexplained dependent variable variance. The value of ρ determines how many terms will be included into the final model and hence the complexity of the model. In this way, the orthogonalization procedure can be implemented in a stepwise manner using backward substitution.

A.3 Conclusions

OLS method is very useful and efficient in model structure selection problem for system identification. Since basis function networks, such as radial basis function networks, wavelet network and neurofuzzy networks, can be expanded in a linear-in-parameter way, the OLS method provides a powerful approach to identify parsimonious model structure of the unknown system.

Appendix B

Affinity Propagation Clustering

In this appendix, we will introduce the affinity propagation (AP) clustering algorithm briefly which is used as a clustering partition method to obtain prior knowledge.

B.1 Introduction

Clustering data by identifying a subset of representative examples is important for detecting patterns in data and in processing sensory signals. Clustering or discovering meaningful partitions of data based on a measure of similarity is a critical step in scientific data analysis and a fundamental problem in computer science. A common approach within the machine learning community involves unsupervised learning of parameters that describe clusters (e.g. the location and scale/shape of the cluster) and partitioning the data by associating every point or region with one or more clusters. In many situations, data is better and more easily characterized by a measure of pairwise similarities rather than defaulting to negative squared Euclidean distance, and in this case, clusters can instead be represented by an "exemplar" data point rather than domain-specific parameters [110, 111]. Data centers, or exemplars, are traditionally found by randomly choosing an initial subset of data points and then iteratively refining it, but this only works well if that initial choice is close to a good solution.

Affinity propagation (AP) is a new algorithm that takes as input measures of similarity between pairs of data points and simultaneously considers all data points as potential exemplars. Real-valued messages are exchanged between data points until a high-quality set of exemplars and corresponding clusters gradually emerges. AP has been used to solve a variety of clustering problems and it uniformly found clusters with much lower error than those found by other methods, and it did so in less than one-hundredth the amount of time. Because of its simplicity, general applicability,

and performance, we believe affinity propagation will prove to be of broad value in science and engineering [112, 113, 114, 115].

B.2 Affinity Propagation Algorithm

AP takes as input a collection of real-valued similarities between data points, where the similarity $s(i,k)$ indicates how well the data point with index k is suited to be the exemplar for data point i. When the goal is to minimize squared error, each similarity is set to a negative squared error (Euclidean distance): For points x_i and x_k, $s(i,k) = -\|x_i - x_k\|^2$. Indeed, the method described here can be applied when the optimization criterion is much more general. When an exemplar-dependent probability model is available, $s(i,k)$ can be set to the log-likelihood of data point i given that its exemplar is point k. Alternatively, when appropriate, similarities may be set by hand.

Rather than requiring that the number of clusters be prespecified, AP takes as input a real number $s(k,k)$ for each data point k so that data points with larger values of $s(k,k)$ are more likely to be chosen as exemplars. As "preferences", these values play important roles to the number of identified exemplars (number of clusters), which is also emerged from the message-passing procedure. AP sends two types of message between data points: responsibility and availability. Responsibilities are sent from data points to candidate exemplars and reflect the evidence of how well-suited the message-receiving point is to serve as an exemplar for the sending point. Availabilities are sent from candidate exemplars to data points and reflect the evidence for how appropriate it would be for the message-sending point to be the exemplar for the message-receiving point. All data points can be considered to be either cluster members or candidate exemplars, depending on whether they are sending or receiving availability or responsibility messages.

Figure B.1 illustrates the two types of messages exchanged between data points. The responsibility $r(i,k)$, sent from data point i to candidate exemplar point k, reflects the accumulated evidence for how well-suited point k is to serve as the exemplar for point i, taking into account other potential exemplars for point i. The availability $a(i,k)$, sent from candidate exemplar point k to point i, reflects the accumulated evidence for how appropriate it would be for point i to choose point k as its exemplar, taking into account the support from other points that point k should be an exemplar.

To begin with, the availabilities are initialized to zero: $a(i,k) = 0$. Then, the parameters are

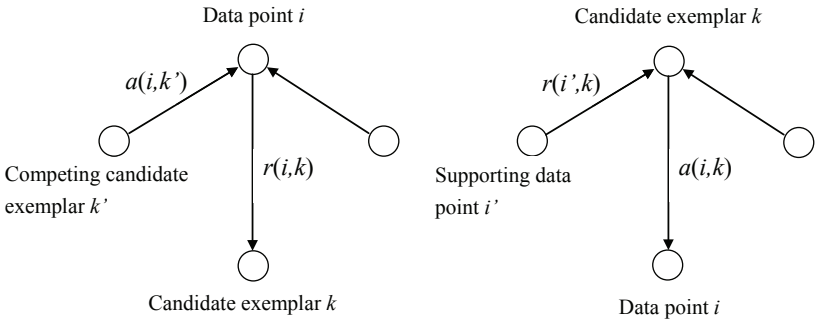


Figure B.1: Two messages are passed between data points: (a)"responsibilities" $r(i,k)$ are sent from data point i to candidate exemplar k; (b) "availabilities" $a(i,k)$ are sent from candidate exemplar k to data point i.

computed and updated using the rules as follows:

$$r(i,k) \leftarrow s(i,k) - \max_{k' \text{ s.t. } k' \neq k} \{a(i,k') + s(i,k')\} \tag{B.2.1}$$

$$a(i,k) \leftarrow \min\left\{0, r(k,k) + \sum_{i' \text{ s.t. } i' \notin \{i,k\}} \max\{0, r(i',k)\}\right\} \tag{B.2.2}$$

$$a(k,k) \leftarrow \sum_{i' \text{ s.t. } i' \neq k} \max\{0, r(i',k)\} \tag{B.2.3}$$

From Eq.(B.2.1), it is known when some points are effectively assigned to other exemplars, their availabilities will drop below zero as prescribed by Eq.(B.2.2). These negative availabilities will decrease the effective values of some of the input similarities $s(i,k')$ in the above rule, removing the corresponding candidate exemplars from competition. The availability $a(i,k)$ is set to the self-responsibility $r(k,k)$ plus the sum of the positive responsibilities candidate exemplar k receives from other points in Eq.(B.2.2). Only the positive portions of incoming responsibilities are added, because it is only necessary for a good exemplar to explain some data points well (positive responsibilities), regardless of how poorly it explains other data points (negative responsibilities). If the self-responsibility $r(k,k)$ is negative (indicating that point k is currently better suited as belonging to another exemplar rather than being an exemplar itself), the availability of point k as an exemplar can be increased if some other points have positive responsibilities for point k being their exemplar. To limit the influence of strong incoming positive responsibilities, the total sum is thresholded

so that it cannot go above zero. However, the self-availability $a(k,k)$ is updated differently as Eq.(B.2.3), where the message reflects accumulated evidence that point k is an exemplar, based on the positive responsibilities sent to candidate exemplar k from other points.

B.3 Conclusions

AP has several advantages over related techniques. Methods such as k-centers clustering, k-means clustering, and the expectation maximization (EM) algorithm store a relatively small set of estimated cluster centers at each step. These techniques are improved upon by methods that begin with a large number of clusters and then prune them, but they still rely on random sampling and make hard pruning decisions that cannot be recovered from. In contrast, by simultaneously considering all data points as candidate centers and gradually identifying clusters, affinity propagation is able to avoid many of the poor solutions caused by unlucky initializations and hard decisions.

As an efficient clustering algorithm with no need to pre-determine the number of clusters, AP is helpful to realize the automatical space partition with outstanding performance.

Appendix C

Multi-objective Evolutionary Algorithm

This appendix presents a brief introduction of the multi-objective evolutionary algorithms (MOEAs), and nondominated sorting genetic algorithm II (NSGA-II) is shown as a representative algorithm in detail.

C.1 Introduction

Over the last two decades, there has been an increasing interest in applying evolutionary algorithms (EAs) to multi-objective optimization problems. This research is highly relevant for real world applications, since real world optimization problems often involve several conflicting objectives for which a tradeoff must be found. The use of EAs to solve problems of this nature has been motivated mainly because of the population-based nature of EAs which allows the generation of several elements of the Pareto optimal set in a single run. Additionally, the complexity of some multi-objective optimization problems (MOPs) may prevent use of traditional techniques [103].

Most MOEAs use Pareto domination to guide the search. A solution s_1 is said to dominate another solution s_2, if is no worse than in all objectives and better than in at least one objective. A solution is said to be nondominated if it is not dominated by any other solution. Ideally, a MOEA returns the Pareto optimal set, the solutions not dominated by any other solution in the search space. In the absence of any further information, one of these Pareto optimal solutions cannot be said to be better than the other. This demands a user to find as many Pareto optimal solutions as possible.

Examples of this work include Corne et al.'s Pareto envelope-based selection algorithm II (PESA-II) [116], Knowles and Corne's Pareto archive evolutionary strategy (PAES) [117], Zitzler and Thiele's strength Pareto evolutionary algorithm (SPEA2) [118], and Deb et al.'s NSGA-II [108]. Common to many MOEAs published these days, they use fitness based on Pareto-domination, which

includes domination counts, nondominated sorting, or identification of the nondominated solutions. What's more, elitism becomes popular and effective in these algorithms since it has been established that preserving the best individuals is important in multi-objective optimization. This is usually accomplished with a huge elite within the population or by an explicit archive separate from the population in which the elite is stored. Another attractive feature of the state-of-art MOEAs is using of niching techniques. In most cases, niching is used as a secondary measure of fitness: If individual s_1 is more nondominated than s_2, s_1 is preferred regardless of niching, while if s_1 and s_2 have the same degree of nondominatedness, the one residing in the most sparsely populated part of the search-space is preferred. As one of the most famous and widely used algorithms, NSGA-II is shown in the following in detail.

C.2 NSGA-II

The NSGA-II, which is an improved version of NSGA [119], is the multi-objective optimization algorithm used in developing the automatic calibration routine. NSGA-II is one of the contemporary multi-objective evolutionary algorithms that exhibits high performance and has been widely applied in various disciplines. The algorithm makes use of a fast non-dominating sorting approach to discriminate solutions, which is based on the concept of Pareto dominance and optimality, thus can be expressed as follows for a multi-objective minimization problem:

$$\begin{aligned} \text{Min} \quad & f(x) = (f_1(x), f_2(x), \cdots, f_n(x)) \\ \text{subject to} \quad & g(x) = (g_1(x), g_2(x), \cdots, g_n(x)) \leq 0 \end{aligned}$$

where $f(x)$ is the vector-valued function, x is the decision vector, and $g(x)$ is a vector of constraints. Considering two decision vectors p and q, p is said to dominate q ($p \prec q$):

$$\begin{aligned} \text{iff} \quad & \forall i \in \{1, 2, \ldots, n\} : f_i(p) \leq f_i(q) \\ & \exists i \in \{1, 2, \ldots, n\} : f_i(p) < f_i(q) \end{aligned}$$

A set of decision vectors that are Pareto optimal within the search space together form a Pareto optimal front. Since NSGA-II is a population-based algorithm, it starts with random generation of parent population, $P(0)$, of potential solutions. The parent population, having size N, is checked for Pareto dominance and a fitness value equal to its non-domination level is assigned to each solution. The non-dominating sorting algorithm uses this fitness value to rank the solutions and assign them to the different fronts (i.e., each solution belongs to different fronts based on its domination level).

The first front contains solutions that dominate solutions of all other fronts. An offspring population of the same size as the parent population is created through recombination based on selection and genetic operators. After the initial generation, the procedure involves comparing the current population with previously identified non-dominated solutions (i.e., elitism). The entire procedure for the t-th generation but the first is described as follows:

1. Combine parent $P(t)$ and offspring $Q(t)$ populations to create R_t of size 2N.
2. Perform non-dominated sorting on R_t to identify the different fronts F_i, where $i = 1, 2, \ldots, l$.
3. Create $P(t+1)$ of size N by selecting solutions from subsequent non-dominated fronts $(F_1, F_2, \cdots, F_l)$.
4. Create $Q(t+1)$ of size N using selection based on crowding-comparison operator, crossover and mutation performed on $P(t+1)$
5. Repeat steps 1 through 4 until convergence criteria have been met.

In the selection of solutions to create $P(t+1)$, when the size of $P(t+1)$ plus F_i is smaller then N, we definitely choose all members of F_i for the new population. The remaining members of the population are chosen from subsequent nondominated fronts in the order of their ranking. However, when the set F_l is the last nondominated set beyond which no other set can be accommodated, and the count of solutions in all sets from F_1 to F_l is larger than the population size N, we choose exactly N population members by sorting solutions in the last front F_l using crowding-distance in descending order and choose the best solutions needed to fill all population slots. The crowding-distance is calculated as average distance between an individual solution and those solutions nearest to it in the objective space, therefore, represents the crowding distance, which is the largest cuboid enclosing that individual solution without including any other solution in the population. The NSGA-II procedure is shown in Fig.C.1.

C.3 Conclusions

As a successfully applied MOEA, NSGA-II can be concluded with following distinctive features:

1. A fast non-dominated sorting procedure is implemented. Sorting the individuals of a given population according to the level of non-domination is a complex task: non-dominated sorting algorithms are in general computationally expensive for large population sizes. The adopted solution performs a clever sorting strategy.

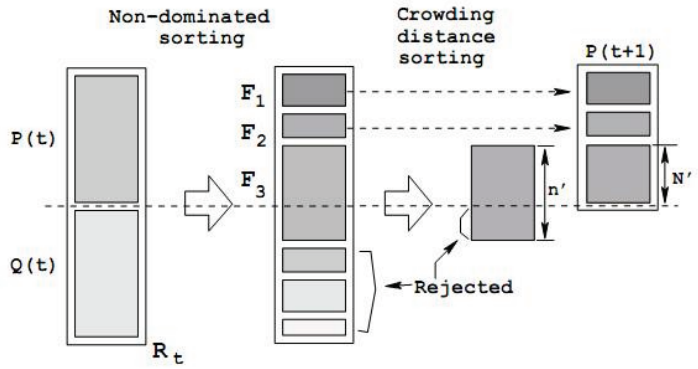


Figure C.1: NSGA-II procedure.

2. NSGA-II implements elitism for multi-objective search, using an elitism-preserving approach. Elitism is introduced storing all non-dominated solutions discovered so far, beginning from the initial population. Elitism enhances the convergence properties towards the true Pareto-optimal set.

3. A parameter-less diversity preservation mechanism is adopted. Diversity and spread of solutions is guaranteed without use of sharing parameters, since NSGA-II adopts a suitable parameter-less niching approach. It is used the crowding distance, which estimates the density of solutions in the objective space, and the crowded comparison operator, which guides the selection process towards a uniformly spread Pareto frontier.

4. The constraint handling method does not make use of penalty parameters. The algorithm implements a modified definition of dominance in order to solve constrained multi-objective problems efficiently.

5. NSGA-II allows both continuous (“real-coded”) and discrete (“binary-coded”) design variables. The original feature is the application of a genetic algorithm in the field of continuous variables.

Bibliography

[1] L. Ljung, *System Identification: Theory for the User, Second Edition*. Prentice-Hall PTR, Upper Saddle River, N.J., 1999.

[2] G. C. Goodwin and K. S. Sin, *Adaptive Filtering Prediction and Control*. Prentice-Hall, Inc., 1984.

[3] L. Ljung, "Identification of nonlinear systems," Department of Electrical Engineering, Linkoping University, SE-581 83 Linköping, Sweden, Tech. Rep. LiTH-ISY-R-2784, 2007.

[4] L. A. Aguirre and C. Letellier, "Modeling nonlinear dynamics and chaos: A review," *Mathematical Problems in Engineering*, vol. 2009, 2009.

[5] L. Ljung, "Perspectives on system identification," *Annual Reviews in Control*, vol. 34, no. 1, pp. 1–12, Mar. 2010.

[6] S. Chen and S. A. Billings, "Neural networks for nonlinear dynamic system modeling and identification," *International Journal of Control*, vol. 56(2), pp. 319–346, 1992.

[7] J. Sjöberg, Q. Zhang, L. Ljung, A. Benveniste, B. Deglon, P. Y. Glorennec, H. Hjalmarsson, and A. Juditsky, "Nonlinear black-box modeling in system identification: a unified overview," *Automatica*, vol. 31(12), pp. 1691–1724, 1995.

[8] J. Sjöberg, H. Hjalmarsson, and L. Ljung, "Neural networks in system identification," Dept of EE. Linköping University, S-581 83 Linköping, Sweden, Tech. Rep. LiTH-ISY-R-1622, 1994.

[9] C. Harris, X. Hong, and Q. Gan, *Adaptive modeling, estimation and fusion from data: A neurofuzzy approach*. Springer Verlag Berlin, 2002.

[10] L.-X. Wang and J. M. Mendel, "Back-propagation fuzzy system as nonlinear dynamic system identifiers," in *Proc. IEEE International Conference on Fuzzy Systems, San Diego, CA, USA*, 1992, pp. 1409–1418.

[11] K. B. Cho and B. H. Wang, "Radial basis function based adaptive fuzzy systems and their applications to system identification and prediction," *Fuzzy Sets and Systems*, vol. 83(3), pp. 325–339, 1996.

[12] J. Park and I. Sandberg, "Approximation and radial-basis-function networks," *Neural Computation*, vol. 5(2), pp. 305–316, 1993.

[13] Y. F. Sun, Y. C. Liang, W. L. Zhang, H. P. Lee, W. Z. Lin, and L. J. Cao, "Optimal partition algorithm of the rbf neural network and its application to financial time series forecasting," *Neural Computing and Applications*, vol. 14(1), pp. 36–44, 2004.

[14] H. Du and N. Zhang, "Time series prediction using evolving radial basis function networks with new encoding scheme," *Neurocomputing*, vol. 71, pp. 1388–1400, 2008.

[15] Q. Zhang and A. Venveniste, "Wavelet networks," *IEEE Transctions on Neural Networks*, vol. 3(6), pp. 889–898, 1992.

[16] Q. Zhang, "Using wavelet network in nonparametric estimation," *IEEE Transctions on Neural Networks*, vol. 8(2), pp. 227–236, 1997.

[17] S. Postalcioglu and Y. Becerikli, "Wavelet networks for nonlinear system modeling," *Neural Computing and Applications*, vol. 16(4-5), pp. 433–441, 2007.

[18] S. Chen, S. A. Billings, and W. Luo, "Orthogonal least squares methods and their application to nonlinear system identification," *International Journal of Control*, vol. 50(5), pp. 1873–1896, 1989.

[19] K. Li, J. X. Peng, and E. W. Bai, "A two-stage algorithm for identification of nonlinear dynamic systems," *Automatica*, vol. 42(7), pp. 1189–1197, 2006.

[20] B. Schölkopf, *Support Vector Learning*. Ph.D dissertation, Technische Universität Berlin, Germany, 1997.

[21] J. Suykens, T. Gestel, J. Brabanter, B. Moor, and J. Vandewalle, *Least Squares Support Vector Machines*. World Scientific, 2002.

[22] D. Mattera and S. Haykin, *Support Vector Machines for Dynamic Reconstruction of a Chaotic System*. Cambridge, MA, MIT Press, 1999.

[23] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support vector machines, regularization, optimization, and beyond*, 2002.

[24] T. Ogunfunmi, *Adaptive nonlinear system identification: the Volterra and Wiener model approaches*. Springer, 2007.

[25] V. Vapnik, *The Nature of Statistical Learning Theory*. Springer Verlag Berlin, 1999.

[26] D. E. Goldberg, *The Design of Innovation: Genetic Algorithms and Evolutionary Computation*. Kluwer Academic, 2002.

[27] W.-D. Chang, "Nonlinear system identification and control using a real-coded genetic algorithm," *Applied Mathematical Modelling*, vol. 31(3), pp. 541–550, 2007.

[28] Q. Chen, K. Worden, P. Peng, and A. Y. T. Leung, "Genetic algorithm with an improved fitness function for (N)ARX modelling," *Mechanical Systems and Signal Processing*, vol. 21(2), pp. 994–1007, 2007.

[29] J. Madár, J. Abonyi, and F. Szeifert, "Genetic programming for the identification of nonliear input-output models," *Industrial & engineering chemistry research*, vol. 44(9), pp. 3178–3186, 2005.

[30] H. L. WEI, S. A. BILLINGS, and J. LIU, "Term and variable selection or non-linear system identification," *International Journal of Control*, vol. 77(1), pp. 86–110, 2004.

[31] S. Chen, X. X. Wang, and C. J. Harris, "NARX-Based nonlinear system identification using orthogonal least squares basis hunting," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 1, pp. 78–84, 2008.

[32] X. Hong, R. J. Mitchell, S. Chen, C. J. Harris, K. Li, and G. W. Irwin, "Model selection approaches for non-linear system identification: a review," *International Journal of Systems and Science*, vol. 39(10), pp. 925–946, 2008.

[33] J. Hu, K. Kumamaru, K. Inoue, and K. Hirasawa, "A hybrid Quasi-ARMAX modeling scheme for identification of nonlinear systems," *Transections of the Society of Instrument and Control Engineers*, vol. 34(8), pp. 997–985, 1998.

[34] J. Hu, K. Kumamaru, and K. Hirasawa, "A Quasi-ARMAX approach to modelling of nonlinear systems," *International Journal of Control*, vol. 74(18), pp. 1754–1766, 2001.

[35] J. Hu and K. Hirasawa, "A method for applying neural networks to control of nonlinear systems," *Neural Information Processing: Research and Development, Springer, Berlin, GERMANY*, pp. 351–369, 2004.

[36] J. Hu, K. Kumamara, K. Inoue, and K. Hirasawa, "KDI-Based robust fault detection in presence of nonlinear undermodeling," *Transections of the Society of Instrument and Control Engineers*, vol. 35(2), pp. 200–207, 1999.

[37] L. Wang, Y. Cheng, and J. Hu, "Adaptive control for nonlinear systems based on quasi-ARX neural network," in *In Proc. of World Congress on Nature and Biologically Inspired Computing (NaBIC2009), Comibatore, India*, 2009, pp. 1548–1551.

[38] ——, "Nonlinear adaptive control using a fuzzy switching mechanism based on improved quasi-ARX neural network," in *In Proc. of the 2010 International Joint Conference on Neural Networks (IJCNN)*, 2010, pp. 3078–3084.

[39] ——, "A quasi-ARX neural network with switching mechanism to adaptive control of nonlinear systems," *SICE Journal of Control, Measurement, and System Integration*, vol. 3(4), 2010.

[40] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Transections on Neural Networks*, vol. 1(1), pp. 4–27, 1990.

[41] M. Norgaard, O. Ravn, N. K. Poulsen, and L. K. Hansen, *Neural Networks for Modeling and Control of Dynamic Systems*. Springer, London, 2000.

[42] T. A. JOHANSEN, "Identification of non-linear systems using empirical data and prior knowledge-an optimization approach," *Automatica*, vol. 32(3), pp. 337–356, 1996.

[43] S. A. BILLINGS and W. S. F. VOONa, "Piecewise linear identification of non-linear systems," *International Journal of Control*, vol. 46(1), pp. 215–235, 1987.

[44] T. A. JOHANSEN and B. A. FOSS, "Constructing NARMAX models using ARMAX models," *International Journal of Control*, vol. 58(5), pp. 1125–1153, 1993.

[45] P.-C. Chang, C.-Y. Fan, and C.-H. Liu, "Integrating a piecewise linear representation method and a neural network model for stock trading points prediction," *IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews,*, vol. 39(1), pp. 80–92, 2009.

[46] J. Hu and K. Hirasawa, "Neural network based prediction model for control of nonlinear systems (in Japanese)," *Trans. of the Society of Instrument and Control Engineers*, vol. 39(2), pp. 166–175, 2003.

[47] J. Hu, X. Lu, and K. Hirasawa, "Training quasi-ARX neural network model by homotopy approach," in *in Proc. of SICE Annual Conference (Hokkaido)*, 2004, pp. 367–372.

[48] H. T. Toivonen, S. Tötterman, and B. Åkesson, "Identification of state-dependent parameter models with support vector regression," *International Journal of Control*, vol. 80(9), pp. 1454–1470, 2007.

[49] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statist.Comput.*, vol. 14, pp. 199–222, 2004.

[50] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation*, vol. 10(5), pp. 1299–1319, 1998.

[51] B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A. J. Smola, "Input space versus feature space in kernel-based methods," *IEEE Transactions on Neural Networks*, vol. 10(5), pp. 1000–1017, 1999.

[52] B. Chen, F. Sun, and J. Hu, "Local linear multi-SVM method for gene function classification," in *Proc. of the World Congress on Nature and Biologically Inspired Computing, Kitakyushu, Japan*, 2010, pp. 183–188.

[53] Y. Cheng, L. Wang, and J. Hu, "A quasi-linear approach for microarray missing value imputation," in *Proc. 2011 International Conference on Neural Information Processing, Shanghai, China*, 2011.

[54] H. H. Szu, B. Telfer, and S. Kadambe, "Neural network adaptive wavelets for signal representation and classification," *Optical Engineering*, vol. 31, pp. 1907–1916, 1992.

[55] Y. C. Pati and P. S. Krishnaprasad, "Analysis and synthesis of feedforward neural networks using discrete affine wavelet transforms," *IEEE Transactions on Neural Networks*, vol. 4(1), pp. 73–85, 1993.

[56] S. A. Billings and H.-L. Wei, "A new class of wavelet networks for nonlinear system identification," *IEEE Transctions on Neural Networks*, vol. 16(4), pp. 862–874, 2005.

[57] Y. Oussar, I. Rivals, L. Personnaz, and G. Dreyfus, "Training wavelet networks for nonlinear dynamic input-output modeling," *Neurocomputing*, vol. 20(1-3), pp. 173–188, 1998.

[58] Y. Oussar and G. Dreyfus, "Initialization by selection for wavelet network training," *Neurocomputing*, vol. 34, pp. 131–143, 2000.

[59] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *science*, vol. 315(5814), pp. 972–976, 2007.

[60] Q. Zhang, "Regressor selection and wavelet network construction," in *Proc. IEEE Conference on Decision and Control (CDC), San Antonio, USA*, 1993.

[61] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[62] Y. Chen, B. Yang, and J. Dong, "Time-series prediction using a local linear wavelet neural network," *Neurocomputing*, vol. 69, pp. 449–465, 2006.

[63] X. Hong and C. J. Harris, "Variable selection algorithm for the construction of MIMO operating point dependent neurofuzzy netoworks," *IEEE Transactions on Fuzzy Systems*, vol. 9(1), pp. 88–101, 2001.

[64] J. Yen and L. Wang, "Simplifying fuzzy rule-based models using orthogonal transformation methods," *IEEE Transections on Systems, Man, and Cybernetics-Part B:Cybernetics*, vol. 29(1), pp. 13–24, 1999.

[65] J. Hu, K. Hirasawa, and K. Kumamara, "A Neurofuzzy-Based adaptive predictor for control of nonlinear systems," *Transections of the Society of Instrument and Control Engineers*, vol. 35(8), pp. 1060–1068, 1999.

[66] R. Šindelář and R. Bakuška, "Input selection for nonlinear regression models," *IEEE Transactions on Fuzzy Systems*, vol. 12(5), pp. 688–696, 2004.

[67] N. Xiong, "Hybrid approach to input selection for complex processes," *IEEE Transections on Systems, Man, and Cybernetics-Part A:Systems and Humans*, vol. 32(4), pp. 532–536, 2002.

[68] L. X. Wang and J. M. Mendel, "Fuzzy basis functions, universal approximation, and orthogonal least-squares learning," *IEEE Transections on Neural Networks*, vol. 3(5), pp. 807–814, 1992.

[69] C. H. North, J. A. Joines, M. G. Kay, C. R. Houck, and C. R. Houck, "A genetic algorithm for function optimization: A matlab implementation," Tech. Rep., 1996.

[70] J. L. Rojo-Álvarez, G. Camps-Valls, M. Martínez-Ramón, E. Soria-Olivas, A. Navia-Vázquez, and A. R. Figueiras-Vidal, "Support vector machines framework for linear signal processing," *Signal Processing*, vol. 85, pp. 2316–2326, 2005.

[71] J. L. Rojo-Álvarez, M. Matínez-Ramón, M. de Prado-Cumplido, A. Artés-Rodríguez, and A. R.Figueiras-Vidal, "Support vector method for robust ARMA system identification," *IEEE Transactions on Signal Processing*, vol. 52(1), pp. 155–164, 2004.

[72] A. Gretton, A. Doucet, R. Herbrich, P. J. W. Rayner, and B. Schölkopf, "Support vector regression for black-box system identification," in *Proc. 11th IEEE Workshop on Statistical Signal Processing*, 2001, pp. 341–344.

[73] Y. Tan and J. Wang, "A support vector machine with a hybrid kernel and minimal vapnik-chervonenkis dimension," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16(4), pp. 385–395, 2004.

[74] M. Szafranski and A. Rakotomamonjy, "Composite kernel learning," in *In Proc. International Conference on Machine Learning, Helsinki, Finland*, 2008.

[75] T. Joachims, N. Cristianini, and J. Shawe-Taylor, "Composite kernels for hypertext categorisation," in *In Proceedings of the International Conference on Machine Learning*, 2001, pp. 250–257.

[76] Z. Barutcuoglu, R. E. Schapire, and O. G. Troyanskaya, "Hierarchical multi-label prediction of gene function," *Bioinformatics*, vol. 22(7), pp. 830–836, 2006.

[77] J. D. Hoheisel, "Microarray technology: beyond transcript profiling and genotype analysis," *Nature Reviews Genetics*, vol. 7, pp. 200–210, 2006.

[78] A. W. C. Liew, N. F. Law, and H. Yan, "Missing value imputation for gene expression data:computational techniques to recover missing data from available information," *Briefings in Bioinformatics*, vol. 12(3), pp. 1–16, 2010.

[79] M. A. Sahu, M. T. Swarnkar, and M. K. Das, "Estimation methods for microarray data with missing values:a review," *International Journal of Computer Science and Information Technologies*, vol. 2(2), pp. 614–620, 2011.

[80] Y. H. Yang, M. J. Buckley, S. Dudoit, and T. P. Speed, "Comparison of methods for image analysis on cDNA microarray data," *Journal of Computational and Graphical Statistics*, vol. 11(1), pp. 108–136, 2002.

[81] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. Altman, "Missing value estimation methods for DNA microarrays," *Bioinformatics*, vol. 17(6), pp. 520–525, 2001.

[82] S. Oba, M. A. Sato, I. Takemasa, M. Monden, K. I. Matsubara, and S. Ishii, "A bayesian missing value estimation method for gene expression profile data," *Bioinformatics*, vol. 19(16), pp. 2088–2096, 2003.

[83] K.-Y. Kim, B.-J. Kim, and G.-S. Yi, "Reuse of imputed data in microarray analysis increases imputation efficiency," *BMC Bioinformatics*, vol. 5(160), 2004.

[84] T. H. Bø, B. Dysvik, and I. Jonassen, "LSimpute: accurate estimation of missing values in microarray data with least squares methods," *Nucleic Acids Res*, vol. 32(3), p. e34, 2004.

[85] H. Kim, G. H. Golub, and H. Park, "Missing value estimation for DNA microarray gene expression data: local least squares imputation," *Bioinformatics*, vol. 21(2), pp. 187–198, 2005.

[86] X. Zhang, X. Song, H. Wang, and H. Zhang, "Sequential local least squares imputation estimating missing value of microarray data," *Computers in Biology and Medicine*, vol. 38(10), pp. 1112–1120, 2008.

[87] Z. Cai, M. Heydari, and G. Lin, "Iterated local least squares microarray missing value imputation," *Journal of Bioinformatics and Computational Biology*, vol. 4(5), pp. 935–957, 2006.

[88] M. Pottmann, H. Unbehauen, and D. E. Seborg, "Application of a general multi-model approach for identification of highly nonlinear processes – a case study," *International Journal of Control*, vol. 57(1), pp. 97–120, 1992.

[89] L. Piroddi and W. Spinelli, "An identification algorithm for polynomial NARX models based on simulation error minimization," *International Journal of Control*, vol. 76(17), pp. 1767–1781, 2003.

[90] A. Sala and C. Ario, "Polynomial fuzzy models for nonlinear control: A taylor series approach," *IEEE Transaction on Fuzzy Systems*, vol. 17(6), pp. 1284–1295, 2009.

[91] E. Hernandez and Y. Arkun, "Control of nonlinear systems using polynomial ARMA models," *AIChE Journal*, vol. 39(3), pp. 446–460, 1993.

[92] R. Haber and H. Unbehauen, "Structure identification of nonlinear dynamic systems – a survey on input/output approaches," *Automatica*, vol. 26(4), pp. 651–677, 1990.

[93] N. Chiras, C. Evans, D. Rees, and M. Solomou, "Nonlinear system modelling: How to estimate the highest significant order," in *In Proc. of IEEE Instrumentation and Measurement Technology Conference, Anchorage, USA*, 2002, pp. 353–358.

[94] S. A. Billings and H. L. Wei, "An adaptive orthogonal search algorithm for model subset selection and non-linear system identification," *International Journal of Control*, vol. 81(5), pp. 714–724, 2008.

[95] Y. Cheng, L. Wang, and J. Hu, "A two-step scheme for polynomial NARX model identification based on MOEA with pre-screening process," *IEEJ Trans. on Electrical and Electronic Engineering*, vol. 6(3), pp. 253–259, 2011.

[96] Y. Cheng, L. Wang, M. Yu, and J. Hu, "An efficient identification scheme for a nonlinear polynomial NARX model," *Artificial Life and Robotics*, vol. 16(1), pp. 70–73, 2011.

[97] H. L. WEI and S. A. BILLINGS, "Feature subset selection and ranking for data dimensionality reduction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29(1), pp. 162–166, 2007.

[98] Y. Cheng, L. Wang, and J. Hu, "A two-step method for nonlinear polynomial model identification based on evolutionary optimization," in *Proc. World Congress on Nature and Biologically Inspired Computing (NaBIC 2009), Coimbatore, India*, 2009, pp. 613–618.

[99] Z. J. Yang, T. Fujimoto, and K. Kumamaru, "A genetic algorithm approach to identification of nonlinear polynomial models," in *Proc. the 12th IFAC Symposium on Identification, Santa Barbara*, 2000.

[100] C. J. Li and Y. C. Yeon, "Genetic algorithm in identifying nonlinear auto regressive with exogenou input models for nonlinear systems," in *In Proc. of the American Control Conference*, 1993, pp. 2305–2309.

[101] A. Shertinsky and R. W. Picard, "On the efficiency of the orthogonal least squares training method for radial basis function networks," *IEEE Transactions on Neural Networks*, vol. 7(1), pp. 195–200, 1996.

[102] H. Shidou, M. Noshiro, Y. Fukuoka, M. Ishikawa, and H. Minamitani, "A method for parameter estimation in the NARMAX model using neural computation," *Transactions of ISCIE*, vol. 9(8), pp. 366–374, 1996.

[103] C. A. C. Coello, G. B. Lamont, and D. A. V. Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems, Second Edition*. Springer, 2007.

[104] K. Rodríguez-Vázquez, C. M. Fonseca, and P. J. Fleming, "Identifying the structure of nonlinear dynamic systems using multiobjective genetic programming," *IEEE Transctions on Systems, Man, and Cybernetics - Part A: Systems AND Humans*, vol. 34(4), pp. 531–545, 2004.

[105] C. Grosan and A. Abraham, "A new approach for solving nonlinear equations systems," *IEEE Transections on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 38(3), pp. 698–714, 2008.

[106] Y. Jin and B. Sendhoff, "Pareto-based multiobjective machine learning: An overview and case studies," *IEEE Transections on Systems, Man, and Cybernetics-Part C: Applications and Reviews*, vol. 38(3), pp. 397–415, 2008.

[107] J. HU, Y. LI, and K. HIRASAWA, "A neural network approach to improving identification of nonlinear polynomial models," in *In Proc. of SICE Annual Conference, Okayama*, 2005, pp. 1662–1667.

[108] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multi-objective genetic algorithm: NSGA-II," *IEEE Transaction on Evolutionary Computation*, vol. 6(2), pp. 182–197, 2002.

[109] X. Hong, M. Brown, S. Chen, and C. J. Harris, "Sparse model identification using orthogonal forward regression with basis pursuit and d-optimality," in *In IEE Proc. of Control Theory and Applications*, 2004, pp. 491–498.

[110] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification, Second Edition*. Wiley-Interscience, 2000.

[111] J. Han, M. Kamber, and J. Pei, *Data mining: concepts and techniques, Second Edition*. Elsevier Inc., 2006.

[112] D. Dueck and B. J. Frey, "Non-metric affinity propagation for unsupervised image categorization," in *In Proc. of International Conference on Computer Vision (ICCV), Rio de Janeiro, Brazil*, 2007.

[113] N. Lazic, I. E. Givoni, P. Aarabi, and B. J. Frey, "FLoSS: Facility location for subspace segmentation," in *In Proc. of International Conference on Computer Vision (ICCV), Kyoto, Japan*, 2009.

[114] N. Lazic, B. J. Frey, and P. Aarabi, "Solving the uncapacitated facility location problem using message passing algorithms," in *In Proc. of International Conference on Artificial Intelligence and Statistics (AISTATS), Sardinia*, 2010.

[115] I. E. Givoni, C. Chung, and B. J. Frey, "Hierarchical affinity propagation," in *In Proc. of Conference on Uncertainty in Artificial Intelligence (UAI), Barcelona, Spain*, 2011.

[116] D. W. Corne, N. R. Jerram, J. D. Knowles, M. J. Oates, and M. J, "PESA-II: Region-based selection in evolutionary multiobjective optimization," in *In Proc. of Genetic and Evolutionary Computation Conference (GECCO)*, 2001, pp. 283–290.

[117] J. D. Knowles and D. W. Corne, "Approximating the nondominated front using the pareto archived evolution strategy," *Evolutionary Computation*, vol. 8(2), pp. 149–172, 2000.

[118] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength pareto evolutionary algorithm," in *In Proc. of EUROGEN2001- Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*, 2001, pp. 95–100.

[119] N. Srinivas and K. Deb, "Multiple objective optimizations using non-dominated sorting in genetic algorithms," *Evolutionary Computation*, vol. 2(2), pp. 221–248, 1994.

Publication List

Journal Paper

1. Yu Cheng, Lan Wang and Jinglu Hu, "Identification of Quasi-ARX Neurofuzzy Model with an SVR and GA Approach", *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E95-A, No. 5*, pp. 876–883, 2012.

2. Yu Cheng, Lan Wang and Jinglu Hu, "An Efficient Identification Scheme for a Nonlinear Polynomial NARX Model", *Artificial Life and Robotics, Vol. 16, No. 1*, pp. 70–73, 2011.

3. Yu Cheng, Lan Wang and Jinglu Hu, "A Two-step Scheme for Polynomial NARX Model Identification Based on MOEA with Pre-screening Process", *IEEJ Transactions on Electrical and Electronic Engineering (TEEE), Vol. 6, No. 3*, pp. 253–259, 2011.

4. Yu Cheng, Lan Wang and Jinglu Hu, "Quasi-ARX Wavelet Network for SVR Based Nonlinear System Identification", *Nonlinear Theory and its Applications (NOLTA), IEICE, Vol. 2, No. 2*, pp. 165–179, 2011.

5. Lan Wang, Yu Cheng and Jinglu Hu, "Stabilizing Switching Adaptive Control for Nonlinear System Based on Quasi-ARX RBFN Model", *IEEJ Transactions on Electrical and Electronic Engineering (TEEE), Vol. 7, No. 4*, in press, 2012.

6. Lan Wang, Yu Cheng and Jinglu Hu, "A Quasi-ARX Neural Network with Switching Mechanism to Adaptive Control of Nonlinear Systems", *SICE Journal of Control, Measurement, and System Integration (SICE JCMSI), Vol. 3, No. 4*, pp. 246–252, 2010.

Conference Paper with Review

7. Yu Cheng and Jinglu Hu, “Nonlinear System Identification Based on SVR with Quasi-linear Kernel”, in *Proc. of 2012 International Joint Conference on Neural Networks (IJCNN)*, Brisbane, Australia, June, 2012, (to appear).

8. Yu Cheng, Lan Wang and Jinglu Hu, “A Quasi-linear Approach for Microaary Missing Value Imputation”, in *Proc. of 18th International Conference on Neural Information Processing (ICONIP’2011)*, pp. 233–240, Shanghai, China, Nov., 2011.

9. Yu Cheng, Lan Wang, Jing Zeng and Jinglu Hu, “Identification of Quasi-ARX Neurofuzzy Model by Using SVR-based Approach with Input Selection”, in *Proc. of 2011 IEEE International Conference on Systems, Man and Cybernetics (SMC’11)*, pp. 1585–1590, Anchorage, U.S., Oct., 2011.

10. Yu Cheng, Miao Yu, Lan Wang and Jinglu Hu, “An Efficient Identification Scheme for Nonlinear Polynomial NARX Model”, in *Proc. of 16th International Symposium on Artificial Life and Robotics (AROB 16th’11)*, pp. 499–502, Beppu, Japan, Jan., 2011.

11. Yu Cheng, Lan Wang and Jinglu Hu, “Quasi-ARX Wavelet Networks for Nonlinear System Identification”, in *Proc. of 2010 International Conference on Modeling, Simulation and Control (ICMSC’10)*, pp. 407–411, Cairo, Egypt, Nov., 2010.

12. Yu Cheng, Lan Wang and Jinglu Hu, “A Two-step Method for Nonlinear Polynomial Model Identification Based on Evolutionary Optimization”, in *Proc. of World Congress on Nature and Biologically Inspired Computing (NaBIC 2009)*, pp. 613–618, Coimbatore, India, Dec., 2009.

13. Yu Cheng, Yongjie Jin and Jinglu Hu, “Adaptive Epsilon Non-dominated Sorting Multi-objective Evolutionary Optimization and Its Application in Shortest Path Problem”, in *Proc. of ICROS-SICE International Joint Conference 2009*, pp. 2545–2549, Fukuoka, Japan, Aug., 2009.