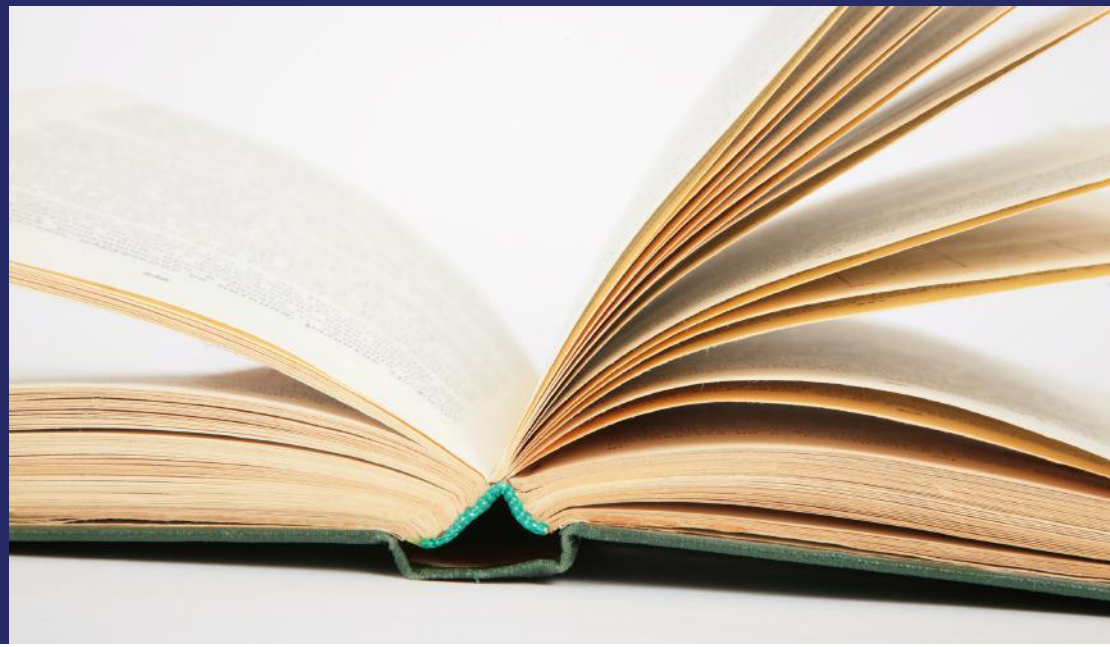
This book contains an identification and modeling system for nonlinear systems under quasi-linear ARX neural network (QARXNN) model. This model is very interesting because it has linear characteristics. Using its parameters, it is easy to derive control system law and to analyze its stability. It has linear and nonlinear parameters, so it can be used to select linear or nonlinear control. We give examples to model and control of nonlinear system under numerical examples and their applications on wind energy conversion systems.

Mohammad Abu Jami'in received the Dr. Eng. in the Graduate School of Information, Production and Systems, Waseda University, Japan 2016. His research interest include artificial intelligence and its application, control system, renewable energy and propulsion systems.

978-613-9-58118-4

Nonlinear Modeling and Control Systems

Jami'in



Mohammad Abu Jami'in

System Identification and Modeling of Nonlinear System

Applied for Control Application

LAP LAMBERT Academic Publishing

Mohammad Abu Jami'in

System Identification and Modeling of Nonlinear System

Applied for Control Application

– Monograph –

February 19, 2018

Omniscriptum Lambert Academic Publishing

Preface

We thank the loving and most Merciful God. This book can be finished by His gift. This book is written according to the field of competence of the author, that is neurocomputing for the identification and design of the control system. Not only the identification of the system using numeric examples, but also applied for the identification of wind energy conversion system and the maximum power point tracking (MPPT) control system. There is no ivory not crack, what is made of human beings there must be defects and disgrace even though done with care. We hope there will be constructive feedback or criticism from our readers and we will welcome it openly and with pleasure. We hope this book will bring benefits to the readers. Finally, the authors hope that this book can inspire the nation's generation to share knowledge through book publishing. Be a dignified, creative, and independent generation.

Surabaya, February 2018

Dr. Eng. Mohammad Abu Jami'in, S.T., M.T.

Contents

1

Introduction

Adaptive control of nonlinear systems had developed significantly during the last few decades. Various adaptive control methods have been proposed, and the global stability and convergence have been proved. Adaptive control of nonlinear systems has also attracted a lot of attention and made significant progress in recent years, along with the introduction of neural networks (NN). The system's input-output is identified by using NN model, hereafter the controller is derived based on the parameters of prediction model. NN based adaptive control can improve control performances because it is generally applicable to the systems with mathematically poorly modeled. However, the main drawback is the lack of systematic design of control methodology. The stability analysis of it is not easy, and the parameter tuning is generally time-consuming process, due to nonlinear and multiparametric nature.

Linear robust adaptive control has been adopted by considering the robustness and performance accuracy. However by increasing the robustness, it will reduce the accuracy of the control system. The use of a nonlinear robust adaptive control offers improved accuracy but can endanger the stability of the closed control system that must be released and switched to the use of a linear robust adaptive controller. To keep both stability and accuracy, a switching technique between linear and nonlinear controller has been proposed.

Nonlinear system with the parameters always uncertain has more than one stability region. Therefore, with only nonlinear controller there is no guarantee that the input-output of closed-loop control becomes bounded. To guarantee bounded control, two controllers is worked with the use of switching mechanism: 1) a linear robust adaptive controller based on linear parameter estimator and 2) a nonlinear robust adaptive controller based on NN model. Two estimators of linear and nonlinear are used under switching mechanism. Linear robust adaptive controller is always stable to ensure the boundedness of the input-output signals of the closed-loop system and also is used to relax the use of nonlinear controller, while nonlinear controller improves the control accuracy. Therefore, by switching mechanism, the control accuracy can be improved and the closed-loop stability can be guaranteed simultaneously.

The control system is generally complex that a system may be linear or nonlinear system under the specific time and conditions. The use of linear models is not sufficient to represent the behavior of the system dynamics which lead to inaccuracies control system itself. While nonlinear models are nonparametric models or multi-parametric that are difficult to analyze and control. Nonlinear models despite having a very good prediction accuracy but it is difficult to guarantee the stability of the closed-loop system because it has many stability regions. Therefore we need a model that can estimate the system with linear and nonlinear approaches simultaneously which will allow for the flexibility of optimization. The QARXNN model has the above specifications, with one model we can estimate linearly or nonlinearly of the system. Thus, it is possible to improve the optimization and derive the control system from the model linearly or nonlinearly.

1.1 System Dynamic

System dynamics is an approach to understanding the behavior of the complex systems over time. For instances, the velocity of car does not change abruptly when the gas pedal is pressed, as well as room's temperature also cannot increase instantly when the AC is turned on. It takes time to produce its effects. Those examples show a dynamic behavior of the system that evolve the time. A feedback system is an effort intended to influence the behavior of system in order to conform to the desired output. The system dynamics is a methodology and mathematical modeling techniques, understanding, and discuss issues with complex problems.

Function approximation is a core of many fields of engineering research such as identification, modeling and simulation. The task of function approximation is to build a function that can represent the dynamics of the input-output (IO) relationship of the system. A usual class of mathematical models for dynamic systems to approximate a function of IO is the ordinary differential equations (ODEs). Mathematically, an ODE written as

$$\frac{dx}{dt} = f(x) \tag{1.1}$$

where $x = [x_1, x_2, \cdots, x_n] \in R^n$ is a vector of real numbers that describes the current state of the system. The equation (1.1) describes the rate of change of the state as a function of the state itself. An example of an ordinary differential equation is a van der Pol equation, which is a model of an electronic oscillator.

$$\begin{aligned} \frac{dx_1}{dt} &= x_1 - x_1^3 - x_2; \\ \frac{dx_2}{dt} &= x_1, \end{aligned} \tag{1.2}$$

The state of the system is presented by two real numbers, x_1 and x_2. The model (1.2) gives the velocity of the state vector for each value of the state.

Along with the dynamic development continued in the 20th century, it has been found that there is a simple dynamic system that is highly sensitive to initial conditions, small disturbances can cause drastic changes in the behavior of the system. The behavior of the system becomes very complicated. The emergence of chaos despite the unique solution has been obtained in determining the initial conditions, in practice it is impossible to make predictions because of the sensitivity of the system to the initial conditions. Differential equation (1.1) is called autonomous systems because there is no external influence. The model is reasonable to be used in celestial mechanics, because it is difficult to influence the movement of the planets.

In many instances, it is useful to model a system that involves the effects of external interference or control force that can be used to control system. One way to capture this is by replacing equation (1.1) with

$$\frac{dx}{dt} = f(x, u) \tag{1.3}$$

where u is a signal from external influences. Model (1.3) is called a forced or controlled differential equation. This model shows that the change of state can be affected by the input, $u(t)$. Adding an input makes model richer and allows new questions to be asked. For example, we can learn what the influence of external disturbances on the trajectory of a system. Or, in the case when the input variables can be modulated, making it possible to steer the system from one point of state to another through proper input selection.

1.2 Identification and Modelling

A model is an exact representation to describe the dynamics of a system used to answer questions through analysis and simulation. With the model allows us to argue about the system and make predictions. The model chosen depends on the questions to be answered, so there are some models allow for a single physical system. In controls, a system can be influenced by external inputs. Space flight is one of the examples where appropriate control inputs required for controlling the orbit. Information also plays an important role in the control because it is very important to know the information about the system by the sensor. The model was modified to include external control forces and sensors. Therefore, the model given by equation (1.3) is replaced with

$$\begin{aligned} \frac{dx}{dt} &= f(x, u), \\ y &= g(x, u), \end{aligned} \tag{1.4}$$

where u is the vector of control signal and y is a vector of measurements. The control system is equipped with sensors and actuators making it possible to obtain a system dynamics model from the experiments. The model

is represented by the data input and output because only these signals are accessible, and it can also be combined with models of physics through the use of feedback and interconnection.

Identification is the determination to the particular class based on the input-output in order to obtain an equivalent tested system. Three elements that need to be involved in the identification: the data, the set of models, and the criterion. The problem posed is to choose the appropriate model and estimate the parameters to adjust the data as best as possible to meet certain criteria [1, 2]. In modeling, there are four issues that need to be studied: 1) the testing procedure, 2) the identification of the model structure, 3) estimation techniques, and initialization requirements. Testing procedures must be able to extract the data needed for model selection and parameter estimation [2]. The structure of model can be described as a selection of model from a recognized model set. The best model is a model that is able to represent the experimental data according to predetermined criteria. There may be many possible model structures. For example, if it is assumed that the system is linear, then the model can be given in terms of the transfer function, impulse response, or as a network structure [1]. Real systems are nonlinear and possibly of infinite dimensions, whereas, to be identified, the members of a model set should be of finite order. Therefore, to assume that the system belongs to the model set is not realistic. The problem is then to identify a model which approximates the data as well as possible. The scheme of identification system shown in Fig. 1.1 is to map IO of experimental data of the dynamic systems.

System identification process is described in Fig. 1.2. Starting with some experts knowledge to the observed system. first of all experiments must be designed with adequate provisions with conventional statistical approach to design experiments. The quality of the received data will greatly affect to the goodness of model. In most cases, post-processing such as signal filtering from the raw data needed for further analysis. If the resulting data does not meet the proposed quality criteria, some details of the experimental design should be improved and steps in the identification procedure must be repeated [3]. A brief overview on classes of system models is given in Fig. 1.3. One key to succeed in system identification based on modelling is to choose an appropriate model type. System theory provides numerous classes of models each with benefits and disadvantages. With high quality data a matching model structure has to be selected. In the following step the model parameters have to be identified. After the identification step, simulations with the developed model of the desired system behaviour can be done and compared to the measured data. If the gap between simulated and measured data is within acceptable limits, a validation should be done. For validation it is important to use a data record, which was not used for parameter identification. If the model does not agree sufficiently with the observed data or is not good enough for the proposed application another iteration beginning with the model selection step must be applied. The model is ready for application if the specific criteria are satisfied [4].

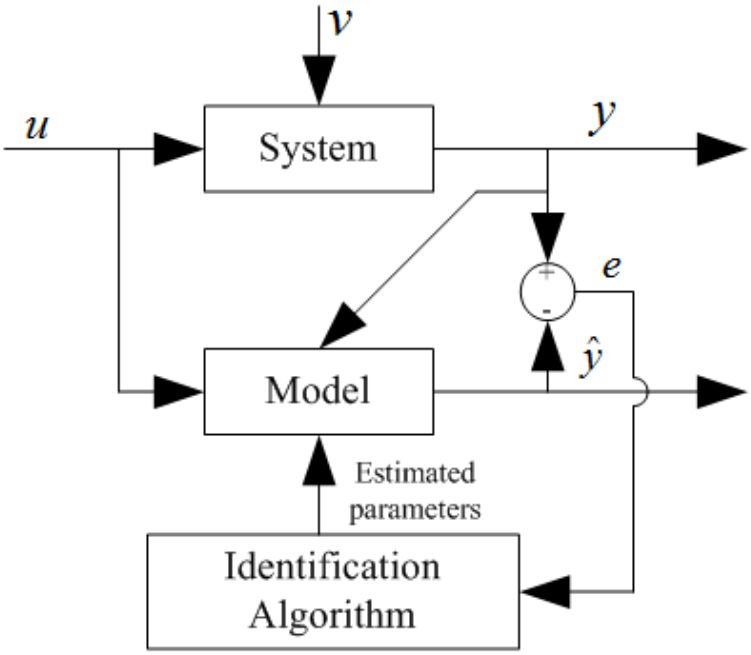


Fig. 1.1. Scheme of identification to estimate parameters.

System identification can be understood as an experimental approach to obtain a mathematical model that reproduces, for desired purposes and with enough exactitude, the dynamic characteristics of the process under study based on the observed (measured) variables of the process: output signal or controlled variable $y(t)$, input signal or control variable $u(t)$, and in some cases disturbances $v(t)$ [5]. Fig. 1.2 shows a general outline of the system identification procedure. In general, this procedure comprises the following stages [5]:

1. Identification experiment design and acquisition of IO data of the process under study. To this end the system must receive an external excitation by applying different input signals and logging the evolution of its input and output signals during a fixed time interval
2. Pre-processing of acquired data. Collected data is generally accompanied by undesired noise and other kind of imperfections which could be necessary to correct before beginning the identification of the process. It aims at preparing data to facilitate and improve the identification.
3. Selection of model structure. If it is desirable to obtain a parametric model, the first step is to determine an appropriate structure for the model. Cer-

tain knowledge about the physical principles that determine the dynamic behaviour of the process greatly facilitates this stage of the process.
4. Estimation of model parameters. As a next step, parameter estimation of the structure that best fits the model response to IO experimental data must be carried out.
5. Model validation. This final stage aims at determining if the obtained model satisfies the exactitude required by the application.

If it is concluded that the obtained model is invalid, the following aspects must be reviewed as possible causes:

- the collected IO data does not provide enough information about system dynamics;
- the chosen structure is not able to describe adequately the model;
- the criteria for parameter adjustment were not properly selected.

1.2.1 Model Structure

Parametric models are described by using a model structure with a finite number of parameters associated to the relevant system signals (input, output and disturbance). These models are described in the discrete time domain, because the identification process is based on experimental data from the sampling results. it is possible to transform a model into continuous time model for further analysis and application. Selection of model structure is one of the most important decisions and difficult to do in a system identification procedure, because the complexity of models that can affect the accuracy in which the model represents the real process [5]. To find the best model implies choosing an appropriate model structure with adequate number of parameters. These structures can be represented by using the following polynomial expression [5, 6]:

$$\hat{y}_{ARX}(k) = \frac{B(q^{-1})}{A(q^{-1})}u(k) + \frac{1}{A(q^{-1})}e(k) \tag{1.5}$$

$$\hat{y}_{OE}(k) = \frac{B(q^{-1})}{F(q^{-1})}u(k) + e(k) \tag{1.6}$$

$$\hat{y}_{ARMAX}(k) = \frac{B(q^{-1})}{A(q^{-1})}u(k) + \frac{C(q^{-1})}{A(q^{-1})}e(k) \tag{1.7}$$

$$\hat{y}_{BJ}(k) = \frac{B(q^{-1})}{F(q^{-1})}u(k) + \frac{C(q^{-1})}{D(q^{-1})}e(k) \tag{1.8}$$

where $\hat{y}_{ARX}(k), \hat{y}_{OE}(k), \hat{y}_{ARMAX}(k), \hat{y}_{BJ}(k)$ are output signals of models with ARX, OE, ARMAX and Box-Jenkins structures, respectively.

Polynomial $A(q^{-1}), B(q^{-1}), C(q^{-1}), D(q^{-1})$ and $F(q^{-1})$ are defined in terms of the backward shift operator q^{-1} and determined as follows:

$$
\begin{aligned}
A(q^{-1}) &= 1 + a_1 q^{-1} + a_2 q^{-2} + ... + a_{na} q^{-na} \\
B(q^{-1}) &= b_0 + b_1 q^{-1} + b_2 q^{-2} + ... + b_{nb} q^{-nb} \\
C(q^{-1}) &= 1 + c_1 q^{-1} + c_2 q^{-2} + ... + c_{nc} q^{-nc} \\
D(q^{-1}) &= 1 + d_1 q^{-1} + d_2 q^{-2} + ... + d_{nd} q^{-nd} \\
F(q^{-1}) &= 1 + f_1 q^{-1} + f_2 q^{-2} + ... + f_{nf} q^{-nf}
\end{aligned}
$$

where na, nb, nc, nd, nf are order of the respective polynomial and a_j, b_j, c_j, d_j, f_j are parameters of the respective model structures that need to be estimated. $e(k)$ is the white noise sequence with zero mean value used to characterize mathematically the dynamic behaviour of disturbances $v(k)$ that act to the system.

Nonlinear system identification has been used in many applications and these include control, instrumentation, power systems, communication systems, nonlinear amplifiers in the transmission stage, signal processing, neuroscience, and in satellite communications, etc. Estimating nonlinear systems is a very broad problem as it is impossible to propose a structure able to describe efficiently every possible nonlinear system [7]. Nonparametric methods of nonlinear system identification do not require parametric expressions for the system nonlinearities. In addition, these methods have the advantage of applicability to nonlinear systems with dynamic nonlinearities [8, 9]. However, these methods take the system as a whole and do not permit separate analysis and identification of the linear dynamics. With the assumption that the nonlinearities in the system are static, in other words, the system dynamics can be expressed in linear terms only, the parametric identification methods can be employed. This assumption leads to nonlinear system structures called the Hammerstein model, the Wiener model and the NARMAX model [8–10].

Neural networks (NN) have been used to identify and control nonlinear dynamical systems because of its ability to approximate arbitrary map to any desired accuracy [11]. The most useful property of neural networks is their ability to approximate arbitrary linear or nonlinear mapping through learning. However, from a user‘s point of view, there are two major criticisms on those neural network models: 1) its parameters do not have useful interpretations and 2) it does not have a friendly interface for controller design and system analysis [12–16]. NN is a multi-parametric model, thus difficult to utilize the parameters of nonlinear system resulted from system identification. To resolve these problems, the idea to model nonlinear system is by using linear relationship between the input vector and its nonlinear coefficient called quasi-linear ARX model. The coefficient is to parameterize the input vector, thus is useful to facilitate control design, to analyze the system using linear theory such as stability with local linear coefficient and to make easy to describe the system by utilizing its coefficient. NN is an embedded system to parameterize input vector. It is called as quasi-linear ARX neural network (QARXNN) shown in Fig. 3.1.

1.2.2 Parameter Estimation

Parameter estimation constitutes a procedure that makes it possible to adjust a model with a specific structure or class of model. For this purpose, it is necessary to determine the estimated parameters of model so as to obtain the model response that best fits IO data obtained during the experiment with a binary signal. This task can be very time consuming indeed. It is very important to choose an appropriate class of model, since its greatly influence the accuracy of model.

Linear estimation models are more consulted in the beginning. Because of their simple structures and well-understood behavior, linear models are widely used in many process modeling. However, the mapping capability of the linear models is usually failed for many noise contaminated system or nonlinear system modeling. As a result, the linear models need to be improved every time. Nonlinear system with the parameters always uncertain, thus the parameter is infinite or multi-parametric. The behavior of the system is much complicated, it is a big difficulty for the identification. Because of wide variety of the systems parameters, there is not a universal solution to identification of systems. Due to nonlinear behavior, in the beginning of the identification task, selection of the identification method is usually the most difficult part. Appropriate methods can be chosen according to system behavior and desired goal of identification. Therefore, it is necessary to use enhanced identification methods to determine the model that approximates the system correctly [17].

The input-output characteristic of the nonlinear systems is changing naturally with high noise disturbance and its time varying behavior. Therefore, the input-output appearance may be linearly or nonlinearly changed. In order to extract these dynamics, a combinatorial models of linear and nonlinear methods are used. To identify the highly nonlinear systems, strong nonlinear models should be used such as neural network, fuzzy logic or their complex recurrent models. These methods do perform highly nonlinear static mapping. However for linear or less nonlinear systems, these nonlinear models are not well suited, resulting in less accurate identification. There should be used linear-nonlinear methods together. Several researchers have already focused on this topic in detail [13, 18–21]. In QARXNN function modeling, parameters are linearly estimated by LSE and residual error of LSE model is parameterized by MLPNN in which linear parameter estimator is set as bias vector of the output nodes of MLPNN. The regressor part of the QARXNN function is the linear and nonlinear functions of the input-output composed with time delay.

1.3 Advance Technology in Model Based Control

The modeling and identification of structural systems with nonlinear behavior constitute a challenging problem in many engineering fields. Efforts have been

devoted to establishing models of nonlinear systems, and various techniques have been developed by many researchers to identify model parameters. The purpose of these system identification methods is to use numerical simulation or field measurements to improve the dynamic modeling capability for prediction, control system, system analysis and stability, fault detection and monitoring, problem optimization, etc. Controllers commanding systems that operate at varying conditions or require high precision operation raise the need for a nonlinear approach in modeling and identification. Most processes in industry are characterized by nonlinear and time-varying behavior. Most systems encountered in the real word are nonlinear in nature, and since linear models cannot capture the rich dynamic behavior of limit cycles, bifurcations, etc. Associated with nonlinear systems, it is imperative to have identification techniques that are specific for nonlinear systems. System identification has become an important area of study because of the increasing need to estimate the behavior of a system with partially known dynamics, especially in the area of control [9, 22–26].

The identification and control of nonlinear dynamical systems has been a challenging problem in the control area for a long time. Since for a dynamic system, the output is a nonlinear function of past output or past input or both, and the exact order of the dynamical systems is often unavailable, the identification and control of this system is much more difficult than that has been done in a static system [27]. To deal with this problem, many soft computing methods such as neural networks and fuzzy neural networks have been developed to process dynamic systems. Neural networks have been widely employed for system identification as they can learn complex mappings from a set of examples. The mapping property, the adaptive nature, and the ability of neural networks to deal with uncertainties make them viable choices for identification and state estimation of nonlinear systems [26].

The input-output approaches to nonlinear system modeling presented are based on a structure which is known as the most common input-output model a nonlinear auto regressive with an exogenous input (ARX) model, which is implemented by using NN. Quasi-ARX model possesses the ARX-linear structure where flexible nonlinear with nonparametric network is applied to interpret the complicated coefficients of the linear structure [14–16, 28]. As a it seems natural to incorporate NN into quasi-ARX model, not only for the flexible capability of approximating, but also better understanding of the network structure and parameters. Moreover, QARXNN predictor could be linear to the input variable $u(t)$, thus the controller can be easily to be derived using its linear inverse and the dynamic behaviour can be analyzed by using approach of linear theory [14, 15, 29]. Due to its simplicity and effectiveness, advanced control strategies such as model-based predictive control and model-based adaptive predictive control under QARXNN model has been developed to solve the complex problem in the control of nonlinear dynamical system [13, 20, 21]. The one of a nonlinear predictive controller based on QARXNN model

is shown by Fig. 1.5. It contains a feedback controller, QARXNN prediction model, and a switching mechanism.

A nonlinear system can be transformed into a linear relationship the the control input by using feedback and coordinate transformation, thus it is called feedback linearizable. Feedback linearization offers methods of building a stabilizing or a tracking controller for a nonlinear system by designing one for the equivalent linear system and utilizing the transformation along with its inverse [30]. With feedback linearization technique, a hybrid control system has been adopted to overcome the system uncertainties that is comprised of an ideal controller and a compensation controller. The ideal controller performed by NN with uncertainty observer is the principle controller, and the compensation controller is designed to compensate the minimum approximation error of the uncertainty observer [31, 32]. With different linearization technique, a quasi-ARX neural network model describe a system is linear to the input controller. The state dependent parameter estimation is nonlinear coefficient used to parameterize the input vector. Therefore, the control law is derived easily by utilizing the transformation by its linear inverse.

Nonlinear system with the parameters always uncertain has more than one stability region. For those problems, with only nonlinear controller there is no guarantee that the input-output of closed-loop control become bounded. To guarantee bounded controller of QARXNN based adaptive control, two controllers were used with switching mechanism: 1) a linear and robust adaptive predictive control based linear parameter estimator, 2) a nonlinear adaptive predictive control based on nonlinear parameters. Two estimators of linear and nonlinear are worked under switching mechanism, the linear robust adaptive predictive control is always stable to ensure the boundedness of the input and output signals of closed-loop system, while nonlinear controller improves the control accuracy. Therefore, by switching mechanism, the control accuracy can be improved and the closed-loop stability can be guaranteed simultaneously.

1.4 Chalanges

Major challenges in modeling based control:

- It may be difficult/time-consuming to obtain empirical models from exhaustive plant tests.
- Controller stability and robustness. Because unstable systems is unusable.
- There is a lack of sensors for key process variables. It can be improved by using state prediction and estimation.
- It is difficult to cope with uncertainties in the real world. It need to create models with uncertainty information, and/or estimate parameters/states on-line and/or use robust optimization techniques.
- Additional future development include decentralized control, hybrid systems, etc.

In many engineering fields, control theory is used to analyze and design feedback loops. Control theory provides a systematic approach to design a closed-loop system to meet the following specifications:

- Ensure that the measured output is equal to (or near) the reference input. the measured output converges to the reference input in the case of regulatory control and disturbance rejection, or the measured output converges to the optimal value in the case of an optimization objective. Accurate systems are essential to ensuring that control objectives are met. For a system in steady state, its inaccuracy or steady state error is the steady state value of the control error $e(k)$.
- Ensure that disturbances acting on the system or nonlinear system do not significantly affect the control output.
- Optimize the control performance index is done by minimization of the cost function index.
- A system has to be stable if for any bounded input, the output is also bounded. Stability is typically the first property considered in designing control systems since unstable systems cannot be used, because it can damage the control system.
- The system has short settling times if it converges quickly to its steady state value.
- The system should achieve its objectives in a manner that does not overshoot. The motivation here is that overshoot typically leads to undershoot and hence to increased variability in the measured output.

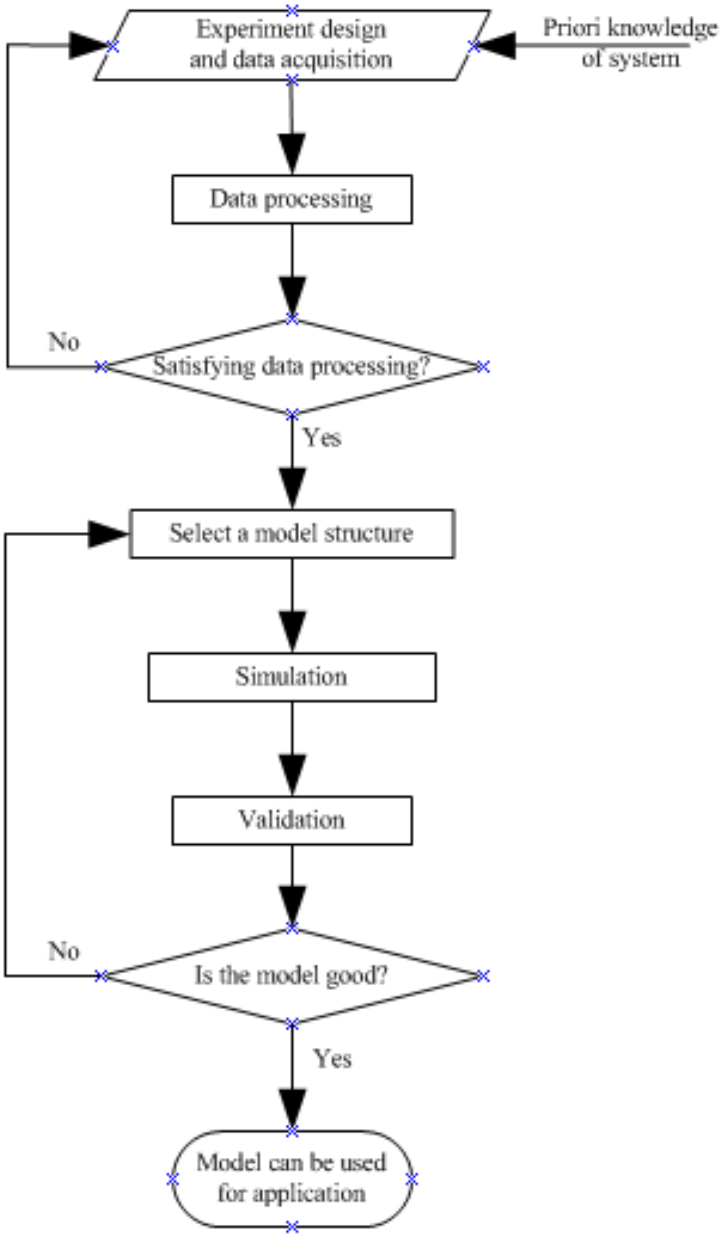


Fig. 1.2. System identification procedure

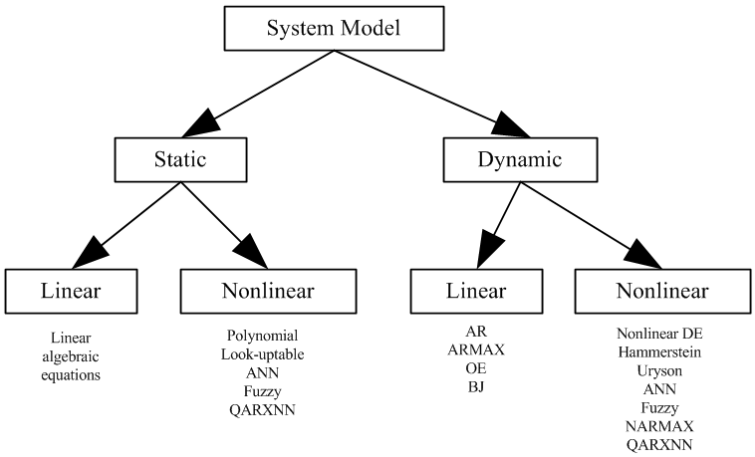


Fig. 1.3. A brief overview on classes of system models

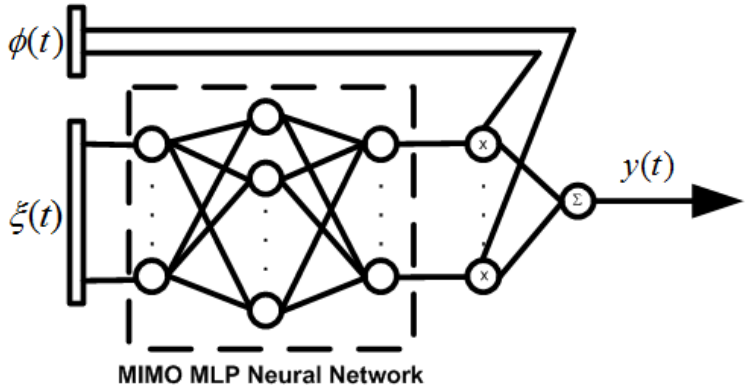


Fig. 1.4. A quasi-linear ARX model with MIMO-MLPNN as an embedded system

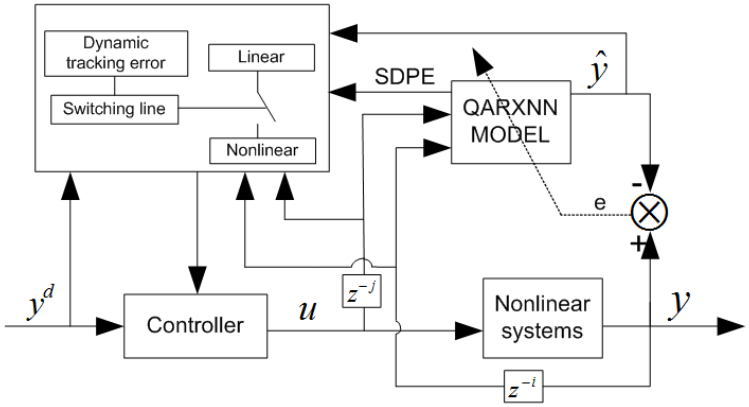


Fig. 1.5. Nonlinear adaptive predictive controller based on QARXNN prediction model. $i = 1, \cdots, n_a$,$j = 1, \cdots, n_b$

2

Lyapunov Learning Approach to Quasi-Linear ARX Model

2.1 Introduction

Nonlinear system identification has been used in many applications such as control, instrumentation, signal processing, satellite communications, power systems, communication systems, neuroscience, and so on. System identification is a broad problem because it is impossible to propose a model able to describe efficiently every possible nonlinear system [7]. Neural networks (NN) and neuro-fuzzy networks are black-box models with nonparametric or multi-parametric models [13, 16, 28]. The most useful property of NN is its ability to approximate arbitrary linear or nonlinear mapping through learning. Black-box methods of nonlinear system identification do not require parametric expressions for the system nonlinearities. In addition, these methods have the advantage of applicability to nonlinear systems with dynamic nonlinearities. However, there are three major drawbacks: 1) these methods take the system as a whole and do not permit separate analysis and identification of the linear dynamics [8–10], 2) the lack of systematic design of control methodology, 3) the stability analysis is not easy, and the parameter tuning is generally time-consuming process, due to its nonlinear and multi-parametric nature [28, 33, 34].

A quasi linear-ARX neural network (QARXNN) is composed hierarchically using linear and nonlinear subsystem modelling. It has two linear and nonlinear sub-models to parameterize the regression vector. Dual mapping performs the coefficient of the regression vector. First, the system is identified using the least square error (LSE) algorithm utilizing the linear sub-model. From the LSE algorithm, we have the estimated parameters of the linear subsystem modeling. Thus, the identified system can be analyzed linearly and the linear adaptive control can be derived. Linear robust adaptive control is able to ensure the closed-loop stability; however, it is low in control accuracy applied to the nonlinear feedback control. Secondly, the estimated linear parameters is set as the bias vector of output nodes for the embedded system performed by NN. The residual error of the linear sub-model is used to train the embedded

system of NN. Thus, the error of the linear sub-model is refined by the nonlinear sub-model. The nonlinear parameters are used to derive the nonlinear robust adaptive control to increase control accuracy. However, it can damage the closed-loop control stability. Therefore, a switching mechanism is used to enhance the control performance. The switching mechanism selects the use of linear and robust adaptive control only when the use of nonlinear and robust adaptive control damages the closed-loop control stability [20, 34, 35].

The QARXNN model is a nonlinear model. It can be simplified as a linear correlation between the regression vector and its coefficients. An embedded system is a subsystem used to give a coefficient for each element of the regression vector. The coefficient is a state-dependent parameter estimation (SDPE) model that consists of two linear and nonlinear sub-models. The linear estimator is performed by least square error (LSE) algorithm, and it is set as a bias vector for the output nodes of NN. The output of NN is nonlinear parameters. The nonlinear parameters of SDPE can be implemented using neuro-fuzzy, wavelet, radial basis function, and multilayer perceptron (MLP) networks [13, 15, 16]. With dual mapping performed hierarchically, if the system identified is linear than SDPE will converge at the fixed value, whereas if the system is nonlinear then SDPE is a time function [36]. In particular applications such as control systems, SDPE is utilized to derive the control law to reduce the computational complexity.

To improve the convergence speed and accuracy, various structure and learning algorithms can be utilized to modify the embedded system of QARXNN. Those systems can be feedforward neural networks with BP training [13], neuro-fuzzy with support vector regression (SVR) and genetic algorithm (GA) [28], and wavelet networks [15]. However, in terms of accuracy and convergence speed, the global minimum point of the cost function of BP algorithm may be trapped in a local minima point. As we know the SVR and GA approach is a hard computation and time consuming process. In addition, The SVR and GA approach mathematically don't discuss about the problem of global minima point of the cost function to the system identification and prediction. Yu et al. [37] have derived a generalized weight update law using a Lyapunov function that guarantees global convergence. However, the update law is of theoretical interest since its implementation will be very much computationally intensive due to the presence of Hessian terms. In another work, Yu et al. [38] have used Lyapunov stability theory to derive a stable learning law for multilayer dynamic neural network. They showed that their learning algorithm is similar to BP algorithm for multilayered perceptron (MLP) with an additional term which ensures the stability of the identification error. In all these works, Lyapunov stability theory has been used to devise learning algorithms to adapt the weights of the network so as to minimize certain cost criteria.

In this paper, an adaptive learning based on the Lyapunov function (ALLF) is proposed. In the proposed structure, a constant learning rate of the BP algorithm is replaced by an adaptive learning rate performed based on the Lyapunov stability theorem. Moreover, the learning method that is

parallel to the BP algorithm makes it easier to use. The coefficients of the adaptive learning are updated every learning cycle based on an energy function, which can obtain a single point global minima [39, 40]. By using adaptive learning based on the Lyapunov function, the feedforward NNs can increase its accuracy, avoid local minima and achieve global minima. Finally, experiments and numerical simulations reveal that the proposed method gives satisfactory results for the system identification of prediction of nonlinear systems.

2.2 Quasi Linear-ARX Model with Lyapunov Learning

Consider a single-input single-output (SISO) black-box time invariant whose the input-output relation described by

$$y(t) = g(\phi(t)) \tag{2.1}$$

where, $g(\cdot)$, $y(t) \in R$, and $\phi(t) = [y(t-1) \cdots y(t-n_y)\, u(t-1) \cdots u(t-n_u)]^T$ are nonlinear function, system output, and a regression vector, respectively. By using Taylor expansion series, nonlinear continuous function $g(\phi(t))$ can be described around the small region of $\phi(t) = 0$,

$$y(t) = g(0) + \dot{g}(0)\phi(t) + \frac{1}{2}\phi^T(t)\ddot{g}(0)\phi(t) + \cdots . \tag{2.2}$$

the equation of (2.2) can be rewritten in matrix equation as

$$y(t) = y(0) + \phi(t)^T\theta(\phi(t)). \tag{2.3}$$

where, $y(0) = g(0)$ is the value for the initial condition and $\theta(\phi(t))$ is the coefficient of the regression vector by

$$\theta(\phi(t)) = [\dot{g}(0)\phi(t) + \frac{1}{2}\phi^T(t)\ddot{g}(0)\phi(t) + \cdots]^T. \tag{2.4}$$

where, $a_{i,t} = a_i(\phi(t)), (i = 1, \cdots, n_y)$ is the coefficient of the regression vector in output and $b_{j,t} = b_j(\phi(t)), (j = 0, \cdots, n_u)$ is is the coefficient of the regression vector in input. Based on the transformation under Taylor series (2.4), we can see that it is infinite coefficients where the components of $[\dot{g}(0)\ddot{g} \cdots]$ are constant. Taylor series only as a bridge to transform nonlinear system. In order to make finite element of the coefficients, we define that $\theta(\phi(t))$ is a nonlinear function where $(\phi(t))$ is an input. Thus, based on the transformation of Taylor series and the system dynamics, nonlinear system can be expressed by [13, 16]:

$$y(t) = \phi^T(t)\aleph(\phi(t)) \tag{2.5}$$

where, $\aleph(\phi(t)) = [a_{(1,t)} \cdots a_{(n_y,t)}\, b_{(1,t)} \cdots b_{(n_u,t)}]^T$ is a nonlinear function injected to the system modelling to parameterize the regression vector of $\phi(t)$.

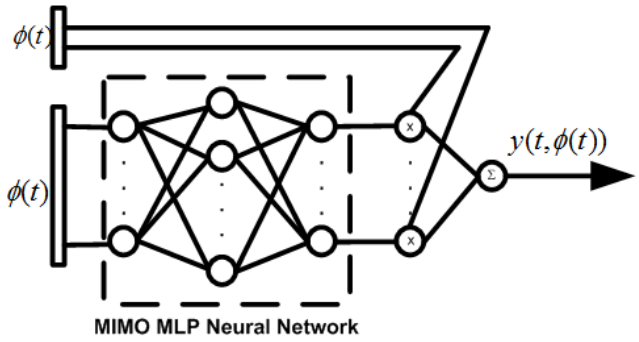


Fig. 2.1. A QARXNN with an embedded MIMO-MLPNN.

n_u and n_y are the orders of time delay for the system input-output [13]. The output of embedded sub-model is state dependent parameter estimation (SDPE), which represents the coefficient of the regression vector.

In this paper, a multi-input multi-output (MIMO) neural networks is used as an embedded sub-model shown by Fig. 3.1, which is described by

$$\aleph(\phi(t), \Omega) = \delta(\phi(t), W) + \theta \tag{2.6}$$

$$\delta(\phi(t), W) = W_2 \Gamma W_1(\phi(t)) \tag{2.7}$$

where $\Omega = \{W_1, W_2, \theta\}$ is the network parameters. Γ, W_1, W_2, and θ are the diagonal nonlinear operator with the identical sigmoidal elements on hidden nodes, the weights of hidden layer, the weights of output layer, and the bias vector for the output nodes, respectively. The structure of the QARXNN model hierarchically is shown by Fig. 2.2. Based on Taylor series expansion, the model is composed of the regression vector and its coefficient shown by (2.5). The identification of the coefficient is performed by sub-model injected into QARXNN model using MIMO neural network. It consists of linear sub-model identified by least square error (LSE) algorithm with the estimated parameters θ and nonlinear sub-model $\delta(\phi(t), W)$ performed by MIMO neural network. θ is set as bias vector for the output nodes of $\aleph(\phi(t), \Omega)$.

In our main theoretical result, the following assumption are made.

A1. The pairs of the input-output of training data are bounded.

A2. The SDPE $\aleph(\phi(t), \Omega)$ are bounded.

A3. There is an optimal weights of SDPE denoted by $\aleph^*(\phi(t), \Omega)$.

By using QARXNN model, the estimated output of (2.5) can be rewritten in the following form:

$$\hat{y}(t) = \phi^T(t)\hat{\aleph}(\phi(t), \Omega) \tag{2.8}$$

if there exists $\aleph^*(\phi(t), \Omega)$ such that

$$y(t) = \phi^T(t)\aleph^*(\phi(t), \Omega) \tag{2.9}$$

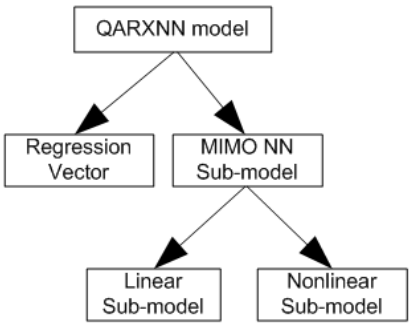


Fig. 2.2. A structure of system modelling of the QARXNN model.

Using (2.8) and (2.9), the adaptation error $E(t)$ can be obtained:

$$\begin{aligned} E(t) &= y(t) - \hat{y}(t) \\ &= \phi^T(t)\aleph^*(\phi(t), \Omega) - \phi^T(t)\hat{\aleph}(\phi(t), \Omega) \\ &= \phi^T(t)\Xi \end{aligned} \tag{2.10}$$

$$\Xi = \aleph^*(\phi(t), \Omega) - \hat{\aleph}(\phi(t), \Omega). \tag{2.11}$$

where Ξ is the adaptation error of SDPE.
Suppose that the assumption **A2.** is valid, then there is a positive real number α for $\| E(t) \| < \alpha$. The regression vector $\phi(t)$ has been fixed. By (2.10) and (2.11), the accuracy and convergence of error characteristic are then greatly influenced by the adaptation error Ξ.

In order to estimate $\aleph(\phi(t), \Omega)$, two sub-models are introduced as:

$$z_l = \phi^T(t)\theta \tag{2.12}$$

$$z_n = \phi^T(t)\delta(\phi(t), W). \tag{2.13}$$

where z_l and z_n are the output of linear sub-model and nonlinear sub-model, respectively. The adaptation error is then distributed to the two sub-models by

$$e_l = y(t) - \phi^T(t)\hat{\theta} \tag{2.14}$$

$$e_n = y(t) - \phi^T(t)\hat{\delta}(\phi(t), W). \tag{2.15}$$

where e_l and e_n are the residual error of the linear and nonlinear sub-models, respectively. Hereinafter, e_l can be set as the reference output for the nonlinear sub-model and e_n is set as the reference output for the linear sub-model, thus the adaptation error can be reduced by dual mapping. By introducing e_l and e_n, equation (2.10) can be rewritten in the following form:

$$\begin{aligned} E(t) &= y(t) - \hat{y}(t) \\ &= y(t) - \phi^T(t)\hat{\aleph}(\phi(t), \Omega) \\ &= y(t) - \phi^T(t)(\hat{\delta}(\phi(t), W) + \hat{\theta}) \\ &= y(t) - \phi^T(t)\hat{\delta}(\phi(t), W) - \phi^T(t)\hat{\theta} \\ &= e_l - z_n \qquad (2.16) \\ &= e_n - z_l. \qquad (2.17) \end{aligned}$$

The equation of (2.12),(2.13),(2.14),(2.15),(2.16) and (2.17) are the direction to train the two sub-models hierarchically.

The desired goal of system identification is to make the adaptation error $E(t)$ goes to zero. Therefore, by (2.16) and (2.17), sub-model z_n is estimated until its output become the same as e_l and vice versa. Two algorithms are implemented hierarchically. Firstly, estimate the system by using linear sub-model under LSE algorithm to obtain θ. Secondly, calculate residual error of linear sub-model by (2.14) and set as the output for nonlinear sub-model performed by multi layer parceptron neural network (MLPNN). By (2.16), If $E(t)$=0 then the output of nonlinear sub-model z_n will converge to e_l.

Nonlinear sub-model in (2.13) is performed by neural networks with $\hat{\theta}$ is a bias vector for the output. The networks weight of NNs is updated based on Lyapubov function defined as follows:

$$V = \frac{1}{2}(e^T e + \lambda \dot{W}^T \dot{W}) \qquad (2.18)$$

where $e = \{e_l^1 - z_n^1, e_l^2 - z_n^2, \cdots, e_l^N - z_n^N\}^T$, N is the number of training data and λ is a positif constant. Performing time derivative of (2.18), we have

$$\begin{aligned} \dot{V} &= \frac{1}{2}(\dot{e}^T e + \frac{1}{2} e^T \dot{e}) + \lambda \ddot{W}^T \dot{W} \\ &= e^T \dot{e} + \lambda \ddot{W}^T \dot{W} \end{aligned}$$

where $\dot{e} = -\frac{\partial z_n}{\partial W}\dot{W} = -H\dot{W}$, then

$$\begin{aligned} \dot{V} &= -e^T \frac{\partial z_n}{\partial W}\dot{W} + \lambda \ddot{W}^T \dot{W}) \\ &= -e^T (H - D)\dot{W} \qquad (2.19) \end{aligned}$$

where $H = \frac{\partial z_n}{\partial W}$ and $D = \lambda \frac{1}{\|e\|^2} e \ddot{W}^T$.

The update trajectory of the weigh vector W is expressed as follows [39]:

$$\dot{W} = \frac{\| e \|^2}{\| H^T e \|^2 + \epsilon} (H - D)^T e \qquad (2.20)$$

where ϵ is a small positive constant. Substituting of (2.20) into (2.19), we have

$$\dot{V} = -\frac{\| e \|^2}{\| H^T e \|^2 + \epsilon} \| (H - D)^T e \|^2 \leq 0 \qquad (2.21)$$

where $(H-D)^T e$ is nonzero, $\dot{V} < 0$ for all $e \neq 0$. $\dot{V}$ is uniformly bounded as $t \to \infty, \dot{V} \to 0$ and $e \to 0$. The weights update law based on ALLF algorithm of (2.20) can be finally expressed in difference equation as follows:

$$\begin{aligned} W(k+1) &= W(k) + \left(\mu \frac{\|e\|^2}{\|H^T e\|^2 + \epsilon} \right) (H-D)^T e \\ &= W(k) + \mu \frac{\| e \|^2}{\| H^T e \|^2 + \epsilon} H^T e \\ &\quad \mu\lambda \frac{\ddot{W}(k)}{\| H^T e \|^2 + \epsilon} \end{aligned} \tag{2.22}$$

where $\ddot{W}(k) = \frac{1}{(\Delta t)^2}[W(k) - 2W(k-1) + W(k-2)]$, μ is a constant coefficient of learning rate selected heuristically, and Δt is taken to be one time unit for simulation.

A hierarchical learning algorithm for QARXNN model under Lyapunov theorem is described as follows:

1. Set $y(t)$ as z_l then estimate θ using LSE algorithm of (2.12).
2. Calculate e_l using (2.14) and set as output guide to train sub-model z_n in (2.13).
3. Use $\hat{\theta}$ as bias vector for output nodes of MLPNN, and small initial values of $W_1(k)$ and $W_2(k)$. Update $W_1(k), W_2(k)$ by (2.13) using adaptive learning based on Lyapunov function. Set $k = 1$, k is the sequence of learning number.
4. Select Lyapunov function candidate, the candidate function is stated as $V = f(e)$, where $V = 0$ only if $e = 0$, $V > 0$ only if $e \neq 0$, $e = e_l - z_n$.
5. Update the learning rate based on energy function by $\dot{V} < 0$. Based on Lyapunov theory, if $V > 0$ and $\dot{V} < 0$, then the error will converge to zero at time goes to infinity $\lim_{k\to\infty} e(k) = 0$.
6. Update the weights using (2.22).
7. Stop if pre-specified condition is met, otherwise goto step 3, set $k = k+1$

If model in (2.5) satisfies to mapping the input-output of the system and the assumption of **A1.**,**A2.** and **A3.** are fulfilled. The prediction output of one ahead prediction is calculated as follows:

$$\hat{y}(t+1) = \phi^T(t+1)\hat{\aleph}(\phi(t+1), \Omega) \tag{2.23}$$

where, $\hat{\aleph}(\phi(t+1), \Omega) = [\hat{a}_{(1,t+1}} \cdots \hat{a}_{(n_y,t+1)} \hat{b}_{(1,t+1)} \cdots \hat{b}_{(n_u,t+1)}]^T$ and $\phi(t+1) = [y(t)\, y(t-1) \cdots y(t+1-n_y)\, u(t)\, u(t-1) \cdots u(t+1-n_u)]^T$ are the estimated of state dependent parameter and the input vector of one step ahead prediction, respectively.

2.3 Experimental Studies

In this section, experiments are performed to clarify the performance of QARXNN based on ALLF algorithm. The proposed identification techniques

are then applied to control of nonlinear systems. The identification parameters are selected as follows: the adaptive learning rate using ALLF algorithm are $\mu = 0.8$ and $\lambda = 0.6$, whereas the constant learning rate of BP algorithm is set at $\eta_{bp} = 0.3$. The update rule of BP is described as follows [41]:

$$\frac{\partial J}{\partial W_2} = -(O_n - z_n)f'(y_{in})\psi_{kj} = -\varphi_k\psi_{kj} \tag{2.24}$$

$$\frac{\partial J}{\partial W_1} = -\sum_k \varphi_k W_2 f'(\psi_{inj})\phi(t) \tag{2.25}$$

$$\triangle W_2 = -\eta_{bp}\frac{\partial J}{\partial W_2} = \eta_{bp}\varphi_k\psi_{kj} \tag{2.26}$$

$$\triangle W_1 = -\eta_{bp}\frac{\partial J}{\partial W_1} = \eta_{bp}\varphi_j\phi(t) \tag{2.27}$$

where $\varphi_k = (O_n - z_n)f'(y_{in})$ and $\varphi_j = \sum_k \varphi_k W_2 f'(\psi_{inj})$. The symbols of y_{in},ψ_{kj},ψ_{inj},η_{bp} denote the input signal for the output nodes, the output signal for the hidden nodes, the input signal for the hidden nodes, and the learning rate of BP algorithm, respectively.

Example 1: System Identification and Prediction of Nonlinear System

Consider a nonlinear dynamic system taken from Narendra [11] that is described as
$y(t) = f(y(t-1), y(t-2), y(t-3), u(t-1), u(t-2))$

$$f(x_1, x_2, x_3, x_4, x_5) = \frac{x_1 x_2 x_3 x_5 (x_3 - 1) + x_4}{1 + x_2^2 + x_3^2}, \tag{2.28}$$

A number of input-output of training data is obtained by testing with pseudo random binary sequence (PRBS) function with the amplitude [-1.0;1.0]. The PRBS signal was generated and is fed to the system (2.28). The first 300 training data are shown in Fig. 2.3. To measure the point of global minima solution, an index of normalized prediction error (NPE) is introduced as follows:

$$NPE = (\frac{\sum_{t=1}^{N}(y(t) - \hat{y}(t))^2}{\sum_{t=1}^{N}(y(t))^2})^{1/2} x100\%. \tag{2.29}$$

The network parameters obtained from the system identification are tested with the input-output of the training data to estimate the one step ahead prediction shown in Fig. 2.4 and Fig. 2.5. As we can see, the NPE index prediction using the proposed method is constant for all time. Thus, a single point global minima is obtainable. To show the effectiveness of faster convergence model of the proposed technique, we also show the performance comparison shown by Fig. 2.6, Fig. 2.7,Fig. 2.8, and Fig. 2.9. We can see, the convergence characteristic of BP is contributed only reducing of error in its learning steps. By using the proposed algorithm the learning convergence

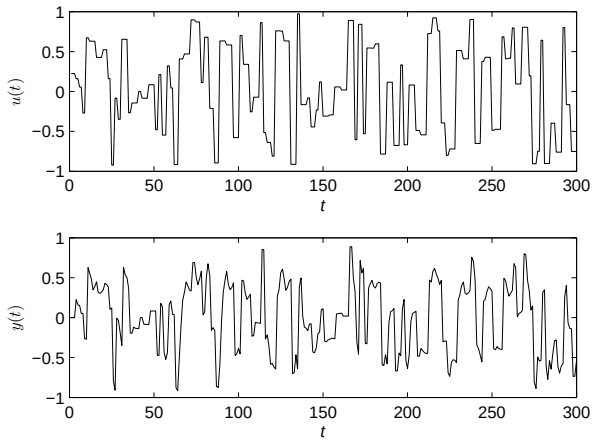


Fig. 2.3. System's input-output of training data.

is optimized by reducing the error and the gradient of error. Based on Lyapunov theorem global minima of system identification can be achieved if both error and gradient of error goes to zero.

The network parameters of the prediction model are also tested with the deterministic signal (2.30). The input vector contains 5 elements as $\phi(t) = [y(t-1)\ y(t-2)\ y(t-3)\ u(t-1)\ u(t-2)]^T$. The QARXNN model with the embedded system of $\aleph_{(5,5,5,1)}$ contains 5 input nodes, 5 hidden nodes and 5 output nodes and one output node. Thus QARXNN model contains 55 parameters.

To test the trained network's model, a set of input-output of test data is generated by the deterministic signal as:

$$u(t) = \begin{cases} \sin(2\pi t/250), & \text{if } t \leq 500 \\ 0.8\sin(2\pi t/250) & \\ +0.2\sin(2\pi t/25), & \text{if } t > 500 \end{cases} \tag{2.30}$$

The performance of prediction model is measured by root mean square (RMS) error index defined as

$$RMS = \sqrt{\frac{\sum_{t=1}^{N}(y(t) - \hat{y}(t))^2}{N}} \tag{2.31}$$

where $\hat{y}(t)$, $y(t)$, $t = 1, 2, \cdots, N$ are the estimated output, the system output, and time. N denotes the length of training data. To demonstrate the performance of prediction model, the results of the trained networks is tested with the deterministic signal (2.30). The testing results are shown in Fig. 2.10 and Fig. 2.12. Compared with the other techniques in [15], the proposed algorithms gives better accuracy as shown in Tab.3.2.

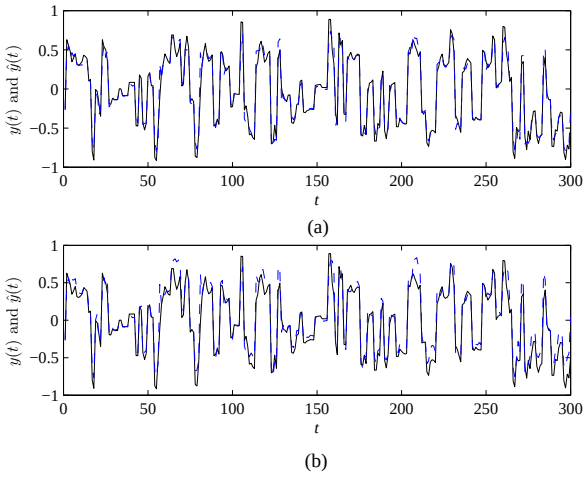


Fig. 2.4. Output of one-step-ahead prediction (solid line: the actual-output, dashed line: prediction-output). (a) the proposed method. (b) the original method [14].

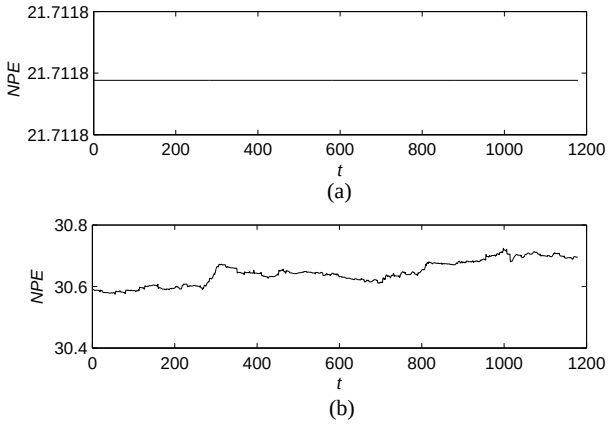


Fig. 2.5. NPE index of one-step-ahead prediction. (a) the proposed method. (b) the original method [14].

2.4 Discussion and Conclusion

This paper presents an adaptive learning based on the Lyapunov function (ALLF) algorithm to train MLPNN embedded on a QARXNN model. There are mainly two contributions: (1) the development of the QARXNN model by using hierarchical adaptive learning for linear and nonlinear sub-models and (2) the adoption of the ALLF algorithm to train the embedded system of MLPNN for the QARXNN model

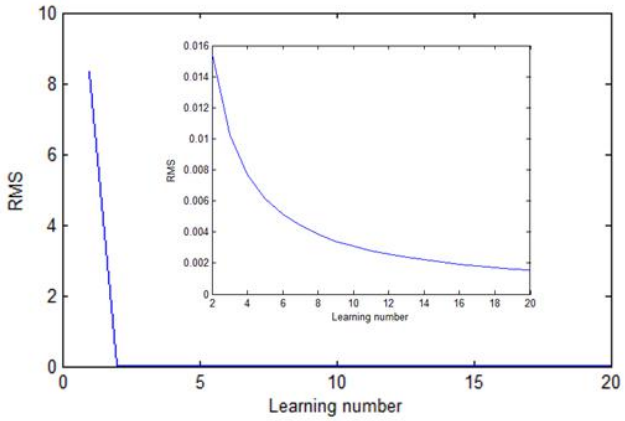


Fig. 2.6. RMS learning of the the proposed method

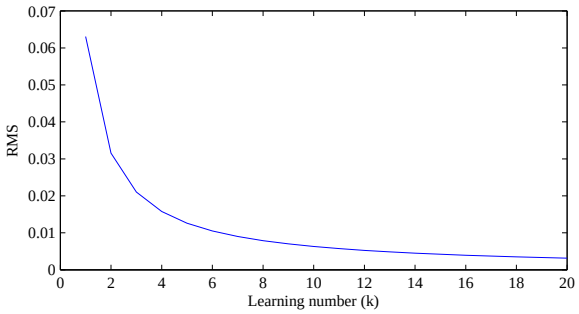


Fig. 2.7. RMS learning using BP method.

Table 2.1. Simulation results and comparison

Model	RMS error	number of parameters
ARX model	0.0866	6
NN	0.0678	341
WN 3	0.0503	12
Q-ARX-NF	0.0478	85
Q-ARX-WN	0.0367	42
Q-ARX-NN	< 0.01	246
Q-ARX-NN-ALLF	0.0297	55

A Lyapunov function based adaptive learning is proposed to train MLPNN of the QARXNN model to obtain global minima and avoid local minima. The coefficients of adaptive learning using the ALLF algorithm are updated every learning cycle based on energy function, which can obtain a single point global minima. Finally, the results of numerical and real system simulation show that

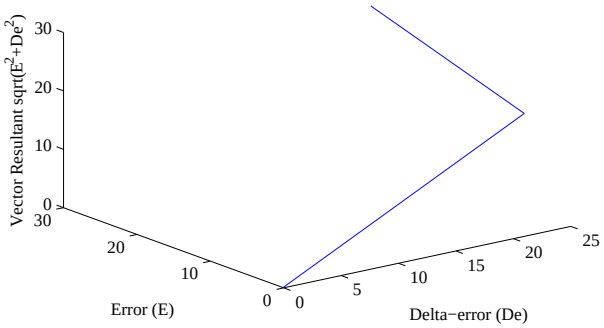


Fig. 2.8. Learning characteristic of the proposed method

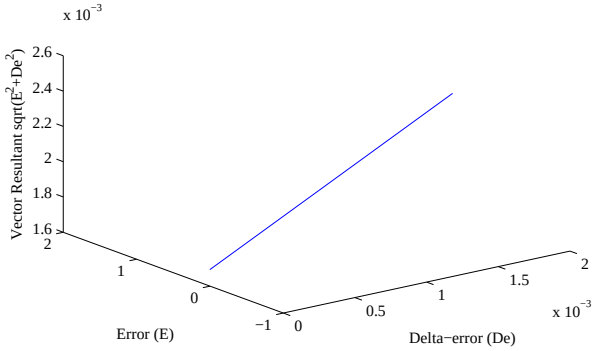


Fig. 2.9. Learning characteristic using BP method

the proposed techniques offer better performance for identification and control of nonlinear dynamical systems.

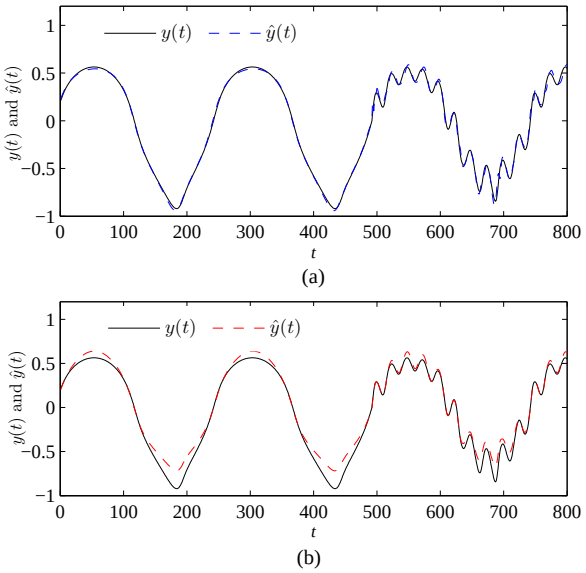


Fig. 2.10. Simulation results with the deterministic signal (a) the proposed method. (b) the original method [14].

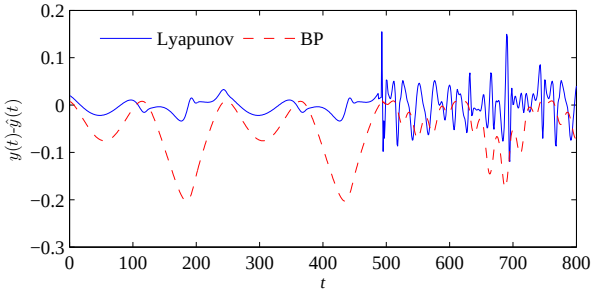


Fig. 2.11. The error of simulation tested with deterministic signal (solid line: the proposed method, dashed line: the original method [14]).

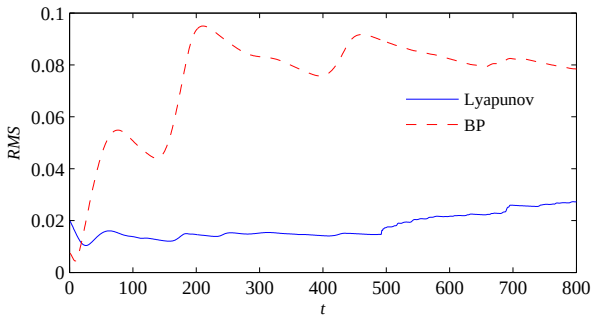


Fig. 2.12. *RMS* errors of the simulation tested with deterministic signal (solid line: the proposed method, dashed line: the original method [14]).

3

Lyapunov Based Switching Control of Quasi-Linear ARX Neural Network Model

3.1 Introduction

If the dynamic model of the controlled system can be known exactly, then the ideal control can be calculated to obtain the desired reference trajectory. Generally, a linear system mixed with noise or nonlinear system results in uncertainty for the parameters of the system. It becomes difficult to solve these problems because the ideal control is unobtainable. Hence, to resolve these problems, a conventional linear and robust control has been adopted to consider the robustness and performance accuracy. However, by increasing the robustness, the control accuracy will be reduced. To maintain the control accuracy, the nonlinear models such as the neural network (NN) and fuzzy models were used as identifiers for the controller design because they are generally applicable to systems with mathematically poor models [20, 42]. An NN-based adaptive control is performed using analysis theory such as stability, robustness, and control accuracy. However, the major disadvantage is the lack of a systematic design of the control methodology [33, 43].

To facilitate the controller-design methodology, the technique of using a feedback linearization of a nonlinear system had been proposed by using a nonlinear model [30, 32, 42–44]. In this paper, a QARXNN model is proposed as an identifier. A QARXNN model is a nonlinear model that describes the system modelling by a linear relationship between the nonlinear coefficients and the regression vector. An NN is used to parameterize the regression vector, with the output being state-dependent parameters estimation (SDPE). Thus, the control law is derived directly by utilizing the transformation of its linear inverse.

It is a key point to guarantee the stability and to improve the control accuracy for designing a control system. The use of a nonlinear controller improves the control accuracy. However, it is difficult to use only a nonlinear controller to guarantee the stability of the closed-loop controller, due to the uncertainty of a nonlinear system [18, 45]. To improve the tracking-control performance, Zhang *et al.* (2010) [18]proposed a switching mechanism to guarantee the sta-

bility and to improve the control accuracy. The switching and tuning framework has been established for the adaptive control design with multiple models [18, 45]. Two linear and nonlinear models are used with a switching mechanism [18]: 1) a linear controller-driven model with self-tuning parameters and 2) an estimator based on an adaptive network-based fuzzy-inference system (ANFIS) for unmodeled dynamics to design a nonlinear controller. Two linear and nonlinear estimators work under a switching mechanism, and the linear adaptive control is always stable to ensure the boundedness of the input and output of the closed-loop system, while the nonlinear controller improves the control accuracy.

The switching index between the linear and nonlinear controller was proposed by some researchers [18–20, 34]. The switching rule is based on the convergence index of error, which is a function of the estimation error of the linear and nonlinear adaptive controllers. It works by comparing the convergence index of the linear and nonlinear parts, which activates the switching mechanism to switch on the controller with the smallest minimum index. However, with such a switching mechanism, it is difficult to obtain more information from the error vector to determine the stability of the closed-loop control system. Therefore, such a switching rule is not efficient, because unnecessary switching to the linear controller will be longer and more frequent. Thus, the accuracy of the control system becomes poor.

In this paper, a new switching mechanism is proposed based on the state of dynamic tracking error so that more information will be provided, not only error but also one up to p-th differential error will be available as the switching variable. The switching mechanism is derived based on Lyapunov stability theorem by utilizing the state parameter of dynamic tracking error obtained from prediction model. Therefore, the switching mechanism become more effective and efficient. Moreover, the proposed switching formula can use the parameter of prediction model presented by SDPE as variables of switching condition criterion.

A QARXNN model can be simplified as a linear correlation between the input vector and its coefficients. An embedded system is a sub-system used to parameterize the regression or input vector. A nonlinear model such as a feedforward NN, a neuro-fuzzy, or a wavelet network can be used as an embedded system. The output is the coefficient of the regression vector called a SDPE [13, 15, 16]. The difference between using an NN as an embedded system for QARXNN models and the others is that the bias vector of the output nodes of an NN is from the estimated parameters of the linear estimator. With the one prediction model of the QARXNN, we have two estimators: linear and nonlinear. A linear parameter estimator (LPE) is estimated using a least-square error (LSE) algorithm. It is set as the bias vector for the output nodes of an NN. The switching mechanism works to select the linear or nonlinear estimators based on the proposed switching-condition index. The controller is derived from the parameter of the selected estimator. The switching-condition index is used only to check the stability condition of a nonlinear controller

based on the Lyapunov Stability Theorem and to switch to the linear controller if it is not stable, and vice versa.

By using the proposed switching law, the controller comprises a linear-robust-adaptive controller (LRAC), a nonlinear-robust-adaptive controller (NRAC), and a switching mechanism. An NRAC controller is designed based on a nonlinear estimator, whereas an LRAC is designed by using a linear estimator. At the beginning, a QARXNN model is used to identify a dynamic system online. The network parameters are updated continuously in accordance with the sampling time. The trained network weights of QARXNN are used to estimate an SDPE by the next regression input. From the estimated parameters of the linear and nonlinear parts, the dynamic tracking error is derived. The stability of the overall system is then verified by the Lyapunov theorem so that ultimately bounded tracking is accomplished.

The main contributions of this paper are summarized as follows. 1) A new controller based on a simplified QARXNN predictive model is constructed to deal with its computational complexity for controlling a nonlinear system mixed with external disturbances. 2) A new switching rule based on the Lyapunov Stability Theorem utilizing the state parameters of dynamic tracking error is proposed so that the controller can work more effectively and efficiently. 3) Simulation results are given to demonstrate the effectiveness of the proposed approach.

3.2 Quasi-ARX Neural Network Model

Consider a single-input, single-output (SISO), black-box, time-invariant system whose input-output relationship is described by the following:

$$y(t) = g(\phi(t)) \tag{3.1}$$

where, $g(\cdot)$, $\phi(t) = [y(t-1)\cdots y(t-n_y)\, u(t-1)\cdots u(t-n_u)]^T$, $y(t) \in R$ is the unknown nonlinear function, regression or input vector, and system output, and $t = 1, 2, \cdots$ denotes the sampling of time. By using a Taylor expansion series and system dynamics, a nonlinear system (3.1) can be presented as a linear correlation between a nonlinear coefficient (Taylor coefficient) and its regression or input vector, described as [13, 16, 20]:

$$y(t) = \phi^T(t)\aleph(\xi(t)). \tag{3.2}$$

where, $\aleph(\xi(t)) = [a_{(1,t)}\cdots a_{(n_y,t)}\; b_{(1,t)}\cdots b_{(n_u,t)}]^T$ denotes the output of an embedded sub-model to parameterize the regression vector. $\xi(t) = [y(t-1)\cdots y(t-n_y)\, u(t-2)\cdots u(t-n_u)\, \nu(t)]^T$ and $\nu(t)$ is the input of an embedded system injected into a QARXNN model and a virtual input, respectively. Incorporated into an NN selected as an embedded system, a QARXNN model is rewritten as:

$$
\begin{aligned}
y(t) &= \phi^T(t)\aleph(\xi(t)) \\
\aleph(\xi(t), \Omega) &= W_2 \Gamma W_1(\xi(t)) + \theta \qquad (3.3) \\
&= \delta(\xi(t)) + \theta \qquad (3.4)
\end{aligned}
$$

where, $\Omega = \{W_1, W_2, \theta\}$ are the network parameters, Γ is the diagonal nonlinear operator with the identical sigmoidal elements on hidden nodes.

In the Equations (3.3) and (3.4), we define the system model with two linear θ and nonlinear $\delta(\xi(t))$ parameters. θ is a bias vector of the output nodes of an NN. The difference with the other NN is that θ is the linear parameters that is estimated based on the linear estimator that uses an LSE algorithm. The coefficient $\aleph(\xi(t))$ of an NN is composed hierarchically based on the following identification scheme: Firstly, the system is estimated under a linear model using a least-square error (LSE) algorithm. Secondly, θ is set as the bias vector of the output nodes of an NN, which is an embedded system of a QARXNN model to parameterize the regression vector. The linear parameter (LP) is estimated by using an LSE algorithm with the output predictor, described by the following:

$$
\begin{aligned}
y_L(t) = {} & a_{(L,1)}y(t-1) + a_{(L,2)}y(t-2) + \\
& a_{(L,n_y)}y(t-n_y) + b_{(L,1)}u(t-1) + \\
& b_{(L,2)}u(t-2) + b_{(L,n_u)}u(t-n_u). \\
y_L(t) = {} & \phi^T(t)\theta \qquad (3.5)
\end{aligned}
$$

where $\theta = [a_{(L,1)} \cdots a_{(L,n_y)}\ b_{(L,1)} \cdots b_{(L,n_u)}]^T$ is the linear-parameter estimation, which is set as a bias vector for MLPNN with. By incorporating $\aleph(\xi(t))$ to ensure the stability and control accuracy, we divide the linear and nonlinear parts of the SDPE equipped with a switching mechanism. When performing a switching mechanism, two linear and nonlinear estimators will be available: 1) a linear estimator with the estimated parameter θ and 2) a nonlinear estimator with the estimated parameter $\aleph(\xi(t))$. The model in (3.2) can be rewritten as follows:

$$
\begin{aligned}
y(t) &= \phi^T(t)(\delta(\xi(t)) + \theta) \\
&= \phi^T(t)\delta(\xi(t)) + \phi^T(t)\theta \qquad (3.6)
\end{aligned}
$$

The details of the algorithm of the quasi-ARX neural-network model can be found in the references [13, 14, 36]. A QARXNN model with an MLPNN set as an embedded system is shown by Fig. 3.1. In our main theoretical result, the following assumption are made:

A1. The pairs of the input-output of the training data are bounded.
A2. The coefficients of the regression vector $\aleph(\xi(t))$ are bounded.
A3. Optimal weights of the regression coefficient $\aleph^*(\xi(t))$ exist.

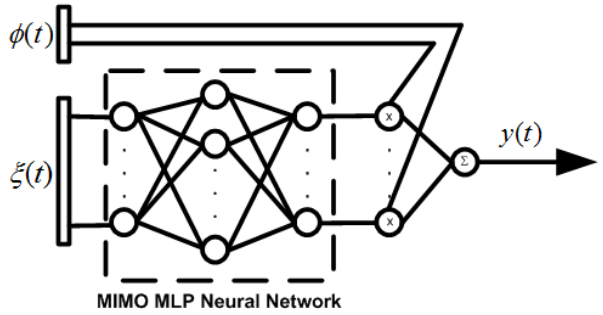


Fig. 3.1. A quasi-ARX neural network model with an embedded system of neural network.

3.3 Control Strategy

A model in (3.2) can be rewritten in the form of the relationship between the input vector and its coefficients as follows:

$$\begin{aligned} y(t) = {} & \hat{a}_{(1,t)}y(t-1) + \hat{a}_{(2,t)}y(t-2) + \\ & \hat{a}_{(n_y,t)}y(t-n_y) + \hat{b}_{(1,t)}u(t-1) + \\ & \hat{b}_{(2,t)}u(t-2) + \hat{b}_{(n_u,t)}u(t-n_u). \end{aligned} \tag{3.7}$$

where, $\hat{\aleph}(\xi(t)) = [\hat{a}_{(1,t)} \cdots \hat{a}_{(n_y,t)}\, \hat{b}_{(1,t)} \cdots \hat{b}_{(n_u,t)}]^T$ is a state-dependent parameter estimation. To derive the control signal, model in (3.7) can be rewritten as

$$u(t-1) = \frac{1}{\hat{b}_{1,t}}(y(t) + g(t)) \tag{3.8}$$

$$\begin{aligned} g(t) = {} & -\hat{a}_{(1,t)}y(t-1) - \hat{a}_{(2,t)}y(t-2) \\ & -\hat{a}_{(n_y,t)}y(t-n_y) - \hat{b}_{(2,t)}u(t-2) \\ & -\hat{b}_{(n_u,t)}u(t-n_u)). \end{aligned} \tag{3.9}$$

If model in (3.2) is rewritten by (3.7), it satisfies the input-output mapping of the system, and the assumptions **A1.**,**A2.** and **A3.** are fulfilled, then the output at time $(t+d)$ can be predicted. Equation (3.2) is regressed at time $(t+d)$ to calculate the output at d steps ahead of the prediction, described as follows:

$$y(t+d) = \phi^T(t+d)\hat{\aleph}(\xi(t+d)) \tag{3.10}$$

where, $\hat{\aleph}(\xi(t+d)) = [\hat{a}_{(1,t+d)} \cdots \hat{a}_{(n_y,t+d)}\hat{b}_{(1,t+d)} \cdots \hat{b}_{(n_u,t+d)}]^T$ is the coefficient of the input vector, $\phi(t+d) = [y(t+d-1)\, y(t+d-2) \cdots y(t+d-n_y)\, u(t+$

$d-1)\, u(t+d-2)\cdots u(t+d-n_u)]^T$ is the input vector at d steps ahead of the prediction, and $\xi(t+d) = [y(t+d-1)\, y(t+d-2)\cdots y(t+d-n_y)\, u(t+d-2)\, u(t+d-3)\cdots u(t+d-n_u-1)\, \nu(t+d)]^T$. The online step ahead of the prediction, d is equal to one. From (3.10), we have the following:

$$u(t) = \frac{1}{\hat{b}_{1,t+1}}(y(t+1) + g(t+1)) \tag{3.11}$$

$$\begin{aligned} g(t+1) = &-\hat{a}_{(1,t+1)}y(t) - \hat{a}_{(2,t+1)}y(t-1) \\ &-\cdots - \hat{a}_{(n_y,t+1)}y(t-n_y+1) - \hat{b}_{(2,t+1)}u(t-1) \\ &-\cdots - \hat{b}_{(n_u,t+1)}u(t-n_u+1)). \end{aligned} \tag{3.12}$$

where, $u(t)$ is a control signal corresponding to a nonlinear estimator by $\hat{\aleph}(\xi(t))$. For the control signal calculated by using a linear predictor, $\hat{\aleph}(\xi(t))$ is replaced with $\hat{\theta}$.

By using a nonlinear estimator, the control accuracy can be maintained. However, the control signal calculated based on a nonlinear estimator is difficult to guarantee the stability of the closed-loop controller. Therefore, a linear estimator is used to keep the closed-loop stability [18, 20, 35]. Thus, the switching line is introduced between the linear part θ and the nonlinear part $\delta(\xi(t))$ of an SDPE, described as follows:

$$\hat{\aleph}(\xi(t)) = \hat{\theta} + \chi(t)\hat{\delta} \tag{3.13}$$

$$u(t) = \chi(t)u_n + (1-\chi(t))u_l(t) \tag{3.14}$$

where, u_l is a control signal calculated by the linear robust control that uses the parameters of the linear estimator $\hat{\theta}$, and u_n is a control signal from the nonlinear robust control that uses the parameters of the nonlinear estimator by summing $\hat{\theta}$ and $\hat{\delta}(\xi(t))$. $\chi(t)$ is a switching line which $\chi(t) = 1$ denoting nonlinear robust control and $\chi(t) = 0$ denoting linear robust control, which is shown by Fig. 3.2.

3.4 Switching Condition

The use of a nonlinear estimator-based control can improve the control accuracy, but it is difficult to ensure the closed-loop stability. The use of a linear estimator-based control can ensure the closed-loop stability, but it is low in accuracy. To improve the overall control performances, a switching condition is used to monitor the stability of the closed-loop system at all times when using a nonlinear controller. Therefore, the analysis of the switching conditions is placed on the use of the nonlinear controller. This proposed switching rule is based on the stability of the dynamic tracking error, defined as follows:

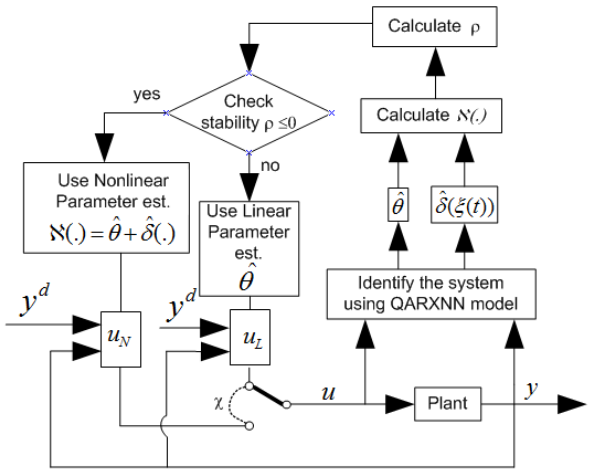


Fig. 3.2. The switching mechanism of the use of the linear and nonlinear parameter estimators

$$\begin{aligned} E(t) &= (e(t), \dot{e}(t), \ddot{e}(t), \cdots, \dot{e}_{p-1}(t)), \\ e(t) &= y(t) - y^d(t), \\ \dot{e}(t) &= \frac{\partial e(t)}{\partial t} = (e(t) - e(t-1))/\Delta t, \\ &\vdots, \\ \dot{e}_{p-1}(t) &= (e(t-p+2) - e(t-p+1))/\Delta t, \end{aligned} \tag{3.15}$$

where, $y^d(t)$ is the reference input trajectory. The tracking error vector is described as follows:

$$\begin{aligned} \dot{e}(t) = \frac{\partial e(t)}{\partial t} &= (e(t) - e(t-1))/\Delta t, \\ &= ((y(t) - y^d(t)) - (y(t-1) - y^d(t-1)))/\Delta t \\ &= (\triangle y(t) - \triangle y^d(t))/\Delta t \simeq \dot{y}(t) - \dot{y}^d(t), \end{aligned} \tag{3.16}$$

where the notation of $\triangle y(t)$ denotes $y(t) - y(t-1)$. The closed-loop system of the tracking error vector dynamics is described as follows:

$$\begin{aligned} \dot{y}_p(t) &= \dot{y}_p^d(t) + K^T E(t) \\ \dot{y}_p(t) - \dot{y}_p^d(t) &= -k_p \dot{e}_{p-1}(t) - k_{p-1}\dot{e}_{p-2}(t) - \cdots - k_1 e(t) \\ \dot{e}_p &= -k_p \dot{e}_{p-1}(t) - k_{p-1}\dot{e}_{p-2}(t) - \cdots - k_1 e(t) \\ 0 &= \dot{e}_p + k_p \dot{e}_{p-1}(t) + k_{p-1}\dot{e}_{p-2}(t) + \cdots + k_1 e(t) \end{aligned} \tag{3.17}$$

where, $K = [k_p, k_{p-1} \cdots k_1] \in R^p$, $k_i (i = 1, \cdots, p)$ are positive constants and p is the degree of tracking error derivative.

We define a nonlinear controller-estimation error as,

$$\begin{aligned} u(t) - u^*(t) &= \frac{1}{\hat{b}_{1,t+1}}(y(t+1) + \hat{g}(t+1)) - \\ &\quad \frac{1}{\hat{b}_{1,t+1}}(y^d(t+1) + g(t+1)) \\ &= \frac{1}{\hat{b}_{1,t+1}}(y(t+1) - y^d(t+1) + \\ &\quad \hat{g}(t+1) - g(t+1)) \\ U(t) &= \frac{1}{\hat{b}_{1,t+1}}(e(t+1) + G) \end{aligned} \tag{3.18}$$

where, $U = u(\cdot) - u^*(\cdot)$,$G = \hat{g}(\cdot) - g(\cdot)$, $\hat{g}(\cdot)$ are calculated using nonlinear predictor of QARXNN model. The error tracking can be obtained as follows:

$$\begin{aligned} e(t+1) &= y(t+1) - y^d(t+1) = \hat{b}_{1,t+1}U(t) - G(t+1) \\ \dot{e}(t+1) &= e(t+1) - e(t) \\ &= \hat{b}_{1,t+1}U(t) - \hat{b}_{1,t}U(t-1) - G(t+1) + G(t) \\ \ddot{e}(t+1) &= \dot{e}(t+1) - \dot{e}(t) \\ &= \hat{b}_{1,t+1}U(t) - 2\hat{b}_{1,t}U(t-1) \\ &\quad + \hat{b}_{1,t-1}U(t-2) - G(t+1) + 2G(t) - G(t-1) \\ \dot{e}_3(t+1) &= \dot{e}_2(t+1) - \dot{e}_2(t) \\ &= \hat{b}_{1,t+1}U(t) - 3\hat{b}_{1,t}U(t-1) \\ &\quad + 3\hat{b}_{1,t-1}U(t-2) - \hat{b}_{1,t-2}U(t-3) \\ &\quad - G(t+1) + 3G(t) - 3G(t-1) + G(t-2) \\ \dot{e}_4(t+1) &= \dot{e}_3(t+1) - \dot{e}_3(t) \\ &= \hat{b}_{1,t+1}U(t) - 4\hat{b}_{1,t}U(t-1) \\ &\quad + 6\hat{b}_{1,t-1}U(t-2) - 4\hat{b}_{1,t-2}U(t-3) \\ &\quad + \hat{b}_{1,t-3}U(t-4) - G(t+1) + 4G(t) \\ &\quad - 6G(t-1) + 4G(t-2) - G(t-3). \end{aligned} \tag{3.19}$$

Through (3.17) and (3.19), the dynamic tracking error can be stated as follows:

$$\dot{E} = AE + BU + G. \tag{3.20}$$

where,

$$A = \begin{pmatrix} 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 1 \\ -k_p & -k_{p-1} & \cdots & -k_1 \end{pmatrix},$$

$$B = \begin{pmatrix} \hat{b}_{1,t+1} & 0 & 0 & 0 \\ \hat{b}_{1,t+1} & -\hat{b}_{1,t} & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ c_1\hat{b}_{1,t+3-p} & -c_2\hat{b}_{1,t+2-p} & \cdots & (-1)^p c_{p+1}\hat{b}_{1,t+2-p} \end{pmatrix},$$

$$U = \begin{pmatrix} U(t) \\ U(t-1) \\ \cdots \\ U(t-p+1) \end{pmatrix}, and$$

$$G = \begin{pmatrix} -G(t+1) \\ -G(t+1) + G(t) \\ \vdots \\ -c_1 G(t+3-p) + \cdots + (-1)^{p+1} c_{p+1} G(t+2-p) \end{pmatrix}$$ where, A is a non-singular matrix and c_n is is a binomial series coefficient such as $\binom{p}{r} = \frac{p!}{r!\,(p-r)!}, 0 \le r \le p$.

By (3.17) and (3.20), we can calculate K such that the roots of the characteristic Equation (3.20) can be chosen strictly in such a way that the poles lie in the left half of the complex plane. This will ensure $\lim_{t\to\infty} e(t) = 0$. A minimum-approximation control error can be defined as follows:

$$\varepsilon = u^* - u(E|\aleph^*(\cdot)). \tag{3.21}$$

The controller's objective is to maintain the stability and accuracy of the closed-loop system by considering ε such that:

$$\aleph^*(\cdot) = \arg\min_{\aleph(\cdot)\in R} [\sup_{E\in R} |U|],$$

where $\aleph^*(\cdot)$ is an optimal network weight that achieves the minimum approximation error obtained through network learning. If the system dynamic in (3.20) is a bounded by ($|U| < \varepsilon$), then there are will be a positive real number of ε. By introducing ε in (3.20), it will be as follows:

$$\dot{E} = AE + B(U(E|\aleph(\cdot)) - U(E|\aleph^*(\cdot)) - \varepsilon) + G. \tag{3.22}$$

Consider a Lyapunov function, stated as follows:

$$V(t) = \frac{1}{2} E^T P E \tag{3.23}$$

where, P is a symmetric positive definite matrix. Since $V(t)$ was selected to be positive definite, $\dot{V}(t)$ has to be a negative semidefinite to make the system uniformly stable. Therefore, we require $\dot{V}(t) = -\dot{E}^T Q E$ to be a negative semidefinite that implies $V(t) \le V(0)$. A negative semidefinite matrix Q is stated as

$$Q = -(A^T P + P A) \tag{3.24}$$

Theorem 1: Suppose a dynamic tracking error is described by the following:

$$\dot{E} = f(E,t) \tag{3.25}$$

where $f(0,t) = 0$ for all instances of t. If there exists a scalar function $V(E,t)$ having a continuous first partial derivative satisfying the conditions:

1. $V(E,t)$ is a positive definite
2. $\dot{V}(E,t)$ is a negative semidefinite,

then the equilibrium state at the origin is uniformly stable. To prove it, we note any trajectory of E such that

$$V(E,t) = V(E,0) + \int_0^t \dot{V}(E,\tau)d\tau. \tag{3.26}$$

$\dot{V}(E,t)$ is a negative semidefinite; hence, $V(E,t)$ is nonincreasing along the corresponding trajectory.

For the system of (3.22), an equilibrium state E_e is defined as $f(E,t) = 0, \forall t$. For nonlinear systems, there are one or more E_e. We denote a spherical region of radius r about an equilibrium state as $\parallel E - E_e \parallel \leq r$ and the Euclidean norm defined by

$$\parallel E - E_e \parallel = ((E_1 - E_{1e})^2 + \cdots + (E_p - E_{pe})^2)^{\frac{1}{2}}. \tag{3.27}$$

Let $S(\gamma)$ consist of all point such that $\parallel E - E_e \parallel \leq \gamma$ where $\gamma \geq \varepsilon$. The time derivative of Lyapunov function along any trajectory is as follows:

$$\begin{aligned}
\dot{V}(t) &= \frac{1}{2}\dot{E}^T PE + \frac{1}{2}E^T P\dot{E}. \\
&= \frac{1}{2}(AE + B(U(E|\aleph(\cdot)) - U(E|\aleph^*(\cdot)) - \varepsilon) + G)^T PE + \\
&\quad \frac{1}{2}E^T P(AE + B(U(E|\aleph(\cdot)) - U(E|\aleph^*(\cdot)) - \varepsilon) + G) \\
&= \frac{1}{2}(E^T A^T PE + E^T PAE) + \frac{1}{2}(B(\tilde{U} - \varepsilon) + G)^T PE + \\
&\quad \frac{1}{2}E^T P(B(\tilde{U} - \varepsilon) + G) \\
&= -\frac{1}{2}(E^T QE) + \frac{1}{2}((B(\tilde{U} - \varepsilon) + G)^T PE \\
&\quad + E^T P(B(\tilde{U} - \varepsilon) + G)) \\
&= -\frac{1}{2}(E^T QE) + (B(\tilde{U} - \varepsilon) + G)^T PE \\
&= -\frac{1}{2}(E^T QE) + (\tilde{U} - \varepsilon)^T B^T PE + G^T PE
\end{aligned} \tag{3.28}$$

where, $\tilde{U} = U(E|\aleph(\cdot)) - U(E|\aleph^*(\cdot))$.

Theorem 2: Using the prediction model of (3.2), the control law given in

(3.14), with the use of nonlinear parameter $\hat{\aleph}(.)$ and a positive constant ε, the switching condition is defined as

$$\rho \leq -\frac{1}{2}(E^T QE) + (\tilde{U} - \varepsilon)^T B^T PE + G^T PE, \rho \leq 0 \tag{3.29}$$

where ρ is a switching condition that is obtained from the time derivative of a Lyapunov function. Therefore, $\lim_{t\to\infty} E(t) = 0$, $E(t) \to 0$ at $t \to \infty$ and the tracking error e will converge to zero.

The switching logic is based on the condition of guaranteeing the stability of the closed-loop controller. The control signal calculated by using nonlinear parameters sometimes breaks the stability of the closed-loop controller $\rho > 0$. We cannot control the unstable system. However, the use of a linear and robust adaptive control(LRAC) is always stable during the whole time. Therefore, to guarantee the stability of the closed-loop controller, an LRAC is used only when the use of a nonlinear and robust adaptive control destroys the stability of the closed-loop controller. According to the Lyapunov theory, the system is stable if the time derivative of a Lyapunov function is negative semidefinite $\rho \leq 0$.

By $\dot{V}(t) \leq 0$, it implies that E is bounded by a positive constant ε that satisfies (3.29). From the convergence analysis based on the Lyapunov theorem, the following can be concluded:

1. $\dot{V}(t)$ is actually a total derivative of $V(t)$ with respect to t along the solution of the system. By $\dot{V}(t) \leq 0$, it implies $V(t)$ is decreasing function of t. By (3.29) with a positive constant ε, the closed-loop error trajectory of (3.23) is a definite positive and nonincreasing, and by (3.22), $\dot{E}$ is also bounded. As a result, the QARXNN-based adaptive control is stable and uniformly bounded. Therefore, $\lim_{t\to\infty} E(t) = 0$, $E(t) \to 0$ at $t \to \infty$, and the tracking error of the closed-loop system e will converge to zero.
2. For linear robust control, A is a nonsingular matrix then there exists one equilibrium state. Therefore, $\dot{V}(t) \leq -\frac{1}{2}(E^T QE), \forall t$ implies $\lim_{t\to\infty} E(t) = 0$, $E(t) \to 0$ at $t \to \infty$, and the tracking error of closed-loop system e will converge to zero for all time.

According **Theorem 2:**, a switching line is used to change control action between linear and nonlinear controllers. The proposed model only with linear parameters has to work until the use of nonlinear parameters does not damage the stability of closed loop system. Therefore, the controller with using linear parameters $\hat{\theta}$ will work all the time, but the nonlinear parameters $\hat{\aleph}(\xi(t))$ will work under the switching sequence. The control law (3.14) works under the switching line as follows:

$$\chi(t) = \begin{cases} 1, \text{ if } \rho \leq 0 \\ 0, \text{ otherwise} \end{cases} \tag{3.30}$$

For the system (3.2), a nonlinear predictive controller based on the QARXNN model contains a feedback controller, a QARXNN predictive model, and a

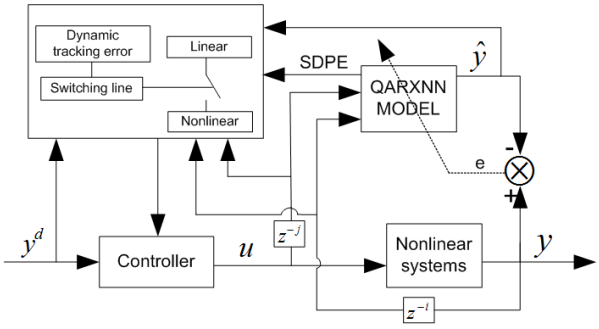


Fig. 3.3. Nonlinear adaptive predictive controller based on QARXNN prediction model.

switching mechanism, as shown by Fig. 3.3. Here, the feedback controller performs based on the dynamic tracking error (3.22) with the Lyapunov Stability Theorem of (3.26) and (3.29). By using a QARXNN prediction model with the two linear and nonlinear estimators (3.5) and (3.2), two controllers perform with the switching mechanism of (3.14).

The switching mechanism selects the use of either the linear or nonlinear controller based on the index of ρ in (3.29) and as presented in Fig. 3.2. By $\rho > 0$, the closed-loop system is unstable using the nonlinear controller. The switching mechanism switches to use the linear controller and reset the nonlinear part $\delta(\xi(t))$ of an SDPE. In the following, the design algorithm of the proposed control law can be summarized as follows:

Step 1. Identify the system under the QARXNN model described in Section 2.

Step 2. Find the estimated parameter of an SDPE using the embedded system of the QARXNN prediction model

Step 3. By using an SDPE, calculate the dynamic tracking error shown by the dynamic matrix of A in (3.20); and by introducing ε, find a new state of dynamic tracking error in (3.22) to obtain stability region with a specific ε ($\varepsilon \leq \gamma$).

Step 4. Check the stability of an NRAC controller by satisfying (3.29) and switching line of (3.30).

Step 5. Calculate the controller signal using (3.11); two controllers can be obtained by using the linear and nonlinear parts parameters of an SDPE via the switching mechanism in (3.14),(3.29) and (3.30).

Step 6. Go to **Step 1.**

3.5 Simulation Results

In this section, two illustrative examples are provided to demonstrate the performance of the proposed APC-QARXNN controller. The examples also show the effect of set-point changes and external disturbances on the control systems employing the proposed controller.

Example 1: Consider the control of a nonlinear discrete-time dynamical system with unstable zerodynamics given by [45] and [18]. The system model is described as follows:

$$\begin{aligned} y(t) &= 2.6y(t-1) - 1.2y(t-2) + u(t-1) \\ &\quad + 1.5u(t-2) + 0.5y(t-1)\mathtt{sin}(u(t-1) \\ &\quad + u(t-2) + y(t-1) + y(t-2)) \end{aligned} \tag{3.31}$$

The objective is to make the system output $y(t)$ track a reference input (desired output trajectory) $y^d(t)$ specified by the following:

$$y^d(t) = 3\mathtt{sign}(\mathtt{sin}(\pi t/50)), \quad 0 < t \leq 250. \tag{3.32}$$

From the system model (3.31), an embedded system MLPNN of QARXNN is constructed with a three-layer neural network. The input vector of $\phi(t)$ is specified by the following: $\phi(t) = [y(t-1)\, y(t-2)\, u(t-1)\, u(t-2)]^T$ and $n_u = 2$ and $n_y = 2$. The number of input nodes, hidden nodes, and output nodes is also the same as $n = n_u + n_y$. The constant learning rate of BP algorithm is selected by $\eta_{bp} = 0.1$ and gain of adaptive tracking control based on the QARXNN model are given by: $\gamma = 0.02$, $p = 2$ and $Q = \begin{pmatrix} 0.1 & 0 \\ 0 & 0.1 \end{pmatrix}$. The output responses, the control signals, the tracking errors, and the switching sequences of the proposed controller compared with the MVC-QARXNN are shown in Fig. 3.4, Fig. 3.5, Fig. 3.6 and Fig. 3.7. To evaluate the performance of the control system, one defines the root-mean-square error (RMS) as follows:

$$RMS = \sqrt{\frac{\sum_{t=1}^{N}(y(t) - y^d(t))^2}{N}} \tag{3.33}$$

where $y^d(t)$, $y(t)$, $t = 1, 2, \cdots, N$ are desired output, the output of controlled system, and the time sampling, and N where is the length of the input-output of controlled system. Fig. 3.7 shows the switching sequence, where $\chi(t) = 1$ denotes the use of NRAC and $\chi(t) = 0$ denotes the use of an LRAC. In Table 3.1, the RMS value of the proposed control system is less than those of the MVC-QARXNN-based control. As we can see, by using the same prediction model, the performance of the proposed controller is significantly better.

As we can see on Fig. 3.7, the use of nonlinear control u_n ($\chi(t) = 1$) is used at almost everywhere in time t. However, linear control u_l is still used to ensure the closed-loop stability. The use only nonlinear control is difficult

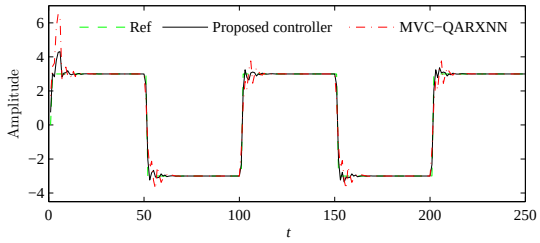


Fig. 3.4. Output responses

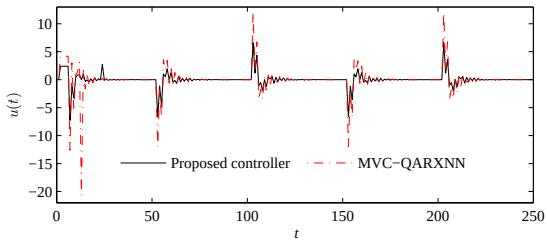


Fig. 3.5. Control signals

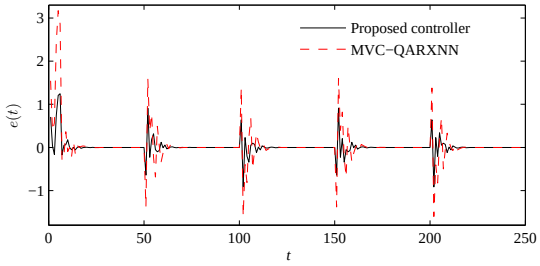


Fig. 3.6. Tracking error

Table 3.1. Simulation results of the control systems

Controllers	Network Parameters	RMS error
Proposed Controller	36	0.201
MVC-QARXNN [20]	36	0.456

to insure the close-loop control stability due to its uncertainty of nonlinear system. We cannot control the unstable system. The result only the use u_n is shown by Fig. 3.8.

Example 2: To further illustrate the applicability of APC-QARXNN in this paper, a nonlinear discrete-time dynamical system mixed with external disturbances given by [46] and [47] is observed. The system model is stated

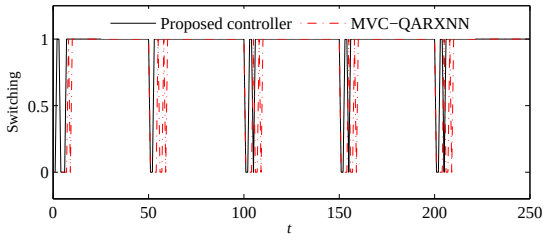


Fig. 3.7. Switching sequence (0: linear; 1: nonlinear)

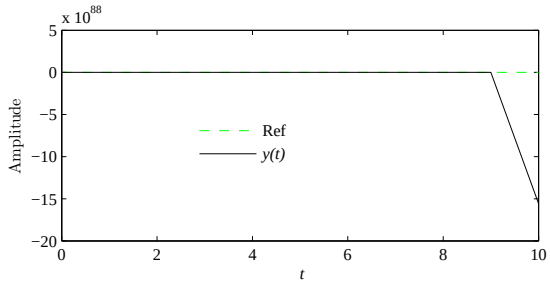


Fig. 3.8. The response of the closed-loop system by using only nonlinear control

as follows:

$$\begin{aligned} y(t) &= 0.9722y(t-1) + 0.378u(t-1) - 0.1295u(t-2) \\ &\quad - 0.3103y(t-1)u(t-1) - 0.04228y^2(t-2) \\ &\quad + 0.1663y(t-2)u(t-2) - 0.03259y^2(t-1)y(t-2) \\ &\quad - 0.3513y^2(t-1)u(t-2) \\ &\quad + 0.3084y(t-1)y(t-2)u(t-2) \\ &\quad + 0.1087y(t-2)u(t-1)u(t-2) + \omega(t). \end{aligned} \tag{3.34}$$

The reference input and the external disturbances $\omega(t)$ are given by

$$y^d(t) = \begin{cases} 1, & 0 < t \leq 200 \\ 0, & 200 < t \leq 400 \end{cases} \tag{3.35}$$

$$\omega(t) = \begin{cases} 0, & 0 < t \leq 100 \\ 0.05, & 100 < t \leq 300 \\ 0.2, & 300 < t \leq 400 \end{cases} \tag{3.36}$$

The input variables of the QARXNN model and the gain of adaptive tracking control are the same as in **Example 1**. To test the robust characteristics of the proposed controller, this example is performed in which the system is mixed with external disturbances. Fig. 3.9, Fig. 3.10, Fig. 3.11, and

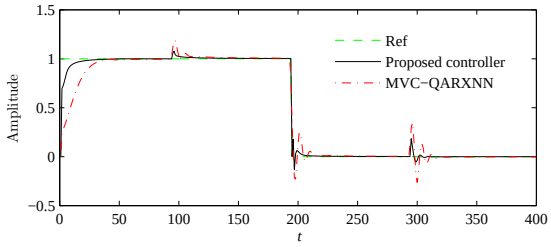


Fig. 3.9. Output responses (under external disturbances)

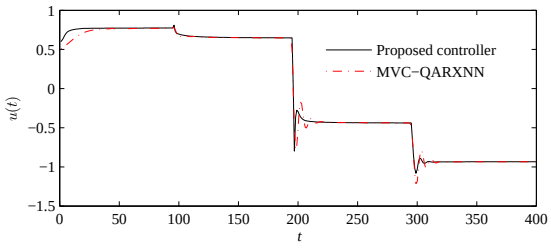


Fig. 3.10. Control signals

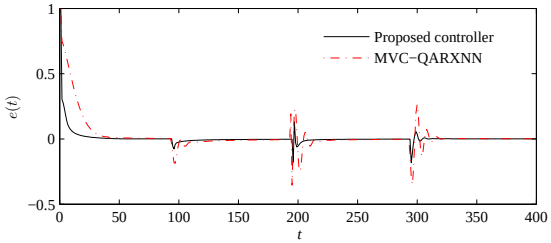


Fig. 3.11. Tracking error

Fig. 3.12 show the output responses, the control signals, the tracking errors, and the switching sequences of the proposed controller compared with the MVC-QARXNN based control. With the output response and error shown by Fig. 3.9 and Fig. 3.11, it indicates that the proposed controller can adapt the external disturbance mixed in nonlinear system. The details of comparison are summarized in Table 3.2. As can be seen, the performance of the proposed controller is better than those that can be obtained with the other controllers.

From the simulation results, Fig. 3.7 and Fig. 3.12 show that the amount of time switching to linear controller is less by using the proposed controller compared to the previous one of MVC-QARXNN-based control. It can be concluded that the proposed switching technique is more effective. The more time switch to nonlinear controller, the accuracy of the control system will be

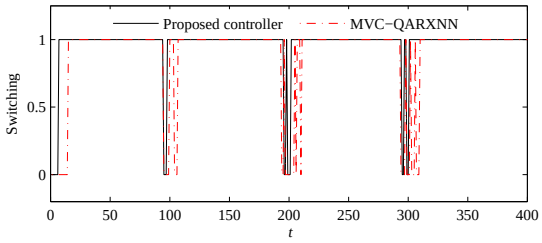


Fig. 3.12. Switching sequence (0: linear; 1: nonlinear)

Table 3.2. Simulation results of the control systems

Controllers	Network Parameters	RMS error
Proposed Controller	36	0.0602
SPC [47]	24	0.0866 *
Fuzzy-based GPC [47]	24	0.1192 *
MVC-QARXNN [20]	36	0.1271
GPC [47]	0	0.1649 *

* The results are listed in the original papers.

increased as well. The use of MVC-QARXNN-based control switch to linear control longer and more often. This is because the switching based on error vector does not provide a lot of information to determine the stability of nonlinear systems. As for switching techniques based on the state of dynamic error, it is possible to get more information about the stability of closed-loop nonlinear controller. Therefore, the controller performance can be increased.

3.6 Discussion and Conclusion

This paper introduces an adaptive controller based on the prediction model of the quasi-ARX neural network (APC-QARXNN). This difference from our previous approach of MVC-QARXNN is shown by its controller strategy and switching rule. The switching based on an index-convergence error is not efficient, because it causes unnecessary switching to the linear controller. Therefore, with APC-QARXNN, a new switching rule based on the Lyapunov Stability Theorem is proposed by utilizing the state parameters of the dynamic tracking error so that the controller can work more effectively and efficiently. The advantages of using an APC-QARXNN are as follows: 1) a simplified quasi-ARX neural-network model presented by a state-dependent parameter estimation (SDPE) is used to derive the controller formulation to deal with its computational complexity. 2) The control law can be derived easily from the model prediction based on the linearization technique, where the system is linear to the input controller. The SDPE is used to parameterize the input

vector. Hence, the control law is derived by utilizing the transformation by its linear inverse. 3) A Lyapunov stability-based switching control is performed to guarantee the closed-loop stability using the state-dependent parameter (SDPE). The proposed switching rule improves the controller's accuracy by reducing unnecessary switching to the linear controller. The major contribution of this paper is the development of the QARXNN-based adaptive control with a new switching mechanism applied for a nonlinear system mixed with external disturbances. Finally, two numerical simulation results confirmed the theoretical analysis.

4

Maximum Power Tracking Control for a Wind Energy Conversion System Based on a Quasi-ARX Neural Network Model

4.1 Introduction

Concerns about both environmental damage caused by burning of fossil fuels and extreme fluctuation in oil prices have aroused worldwide interest in using alternative energy sources [48, 49]. Among these sources, wind energy is a very promising and abundantly available alternative. Clean renewable energy sources (i.e., wind, photovoltaic [PV], and fuel cells) do not cause global warming, and according to estimates from the European Wind Energy Association, tapping only 10% of available wind power can sufficiently meet the world's total electricity needs. Given current technological advances in the field of power electronics, variable speed drives, and wind turbines, the cost of wind power can approach the cost of maintaining fossil-fuel power plants. The US and Germany are world leaders in terms of using wind energy's installed capacity (25 GW), while in terms of the use of wind energy, Denmark reports the highest percentage, which is 20% of total energy use. Though wind energy currently only accounts for 1 of all generated electricity, this figure is expected to rise to 20% by 2030 [49].

The control system plays an important role in extracting maximum energy from wind power. The scheme of maximum power point tracking (MPPT) control is reported to operate by varying the speed of the generator so that the turbine can operate at the point of maximum aerodynamic efficiency and, in turn, obtain maximum power extracted from wind [50]. The amount of power that can be converted depends on the MPPT accuracy, which is highly influenced by the accuracy of the control system. The effectiveness of the controller is intended to maximize the power output of the turbine, regardless of the type of generator used.

The issues involved in the design of a nonlinear controller for MPPT concern the robustness of parameter uncertainty since the parameters are always subject to change according to the change of wind speed [51]. In conventional linear controls for robustness, increasing robustness can reduce the accuracy of the control system [13, 20]. To this end, several approaches have been devel-

oped to solve the control problems associated with wind energy. An adaptive neural network model-based estimator has been developed to estimate uncertain aerodynamics online, after which a tracking control law can be derived based on Lyapunov stability analysis [52]. The wind energy conversion system (WECS) can also be modeled as a linear system by using a linear parameter estimator (LPE), in which the controller uses a hybrid control unit of the linear quadratic Gaussian (LQG) and neurocontroller (NC) performed by a feed-forward radial basis function (RBF) model [53]. However, as we know, fuzzy systems, neural networks, and neurofuzzy systems are black-box models. The stability analysis of these models is difficult, and parameter tuning is generally a time-consuming process due to its nonlinear and multiparametric nature [28, 33]. Among other drawbacks, LQG controllers designed with an LPE estimator also require additional estimation schemes using neurocontrol for systems with parameter uncertainties. In this paper, we propose a QARXNN prediction model to estimate parameters of the input vector, and by performing the minimum variance control law, the estimated parameter is set as the controller parameters with switching law [13].

With QARXNN, we assume that WECS is a nonlinear system in which nonlinearity is placed on the parameter estimates. The estimated parameters have a linear relationship with the regression vector that makes easy to derive control law from the proposed model. The linear part parameters are used in the whole time, while the nonlinear part parameters working under switching function. The use of nonlinear parameters can improve the accuracy of control, but sometimes damage control system, so that only the linear parameters are working. Nonlinear parameters will work until the system recovers. To begin, a QARXNN model is used to identify a dynamic system online. The network parameters are updated continuously in accordance with the sampling time. The trained network weights of QARXNN is used to estimate linear and nonlinear parameters by the next regression input. By using Lyapunov based switching control under QARXN model, we derive the controller law using linear and nonlinear parts of model then the controller signal is calculated by using estimated linear and nonlinear parts. The stability of the closed-loop control system is maintained by switching linear and nonlinear parts; switching to the nonlinear part maintains accuracy, while switching to the linear part guarantees stability. Though a QARXNN model is built by using a neural network, the parameters obtained can be simplified in the form of regression coefficients. Control signals can be constructed directly by using regression coefficients.

By performing Taylor series expansions, a nonlinear system develops a linear correlation between the regression vector and its coefficients [13]. The coefficients serve as the parameters of the input vector called kernel functions that can be executed by using a multi-input multi-output (MIMO) model. These also can be executed by neurofuzzy, wavelet, radial basis function, and MLPNN [15]. The accuracy, stability, and the speed of convergence can be improved with Lyapunov training [16]. The QARXNN can also be used to

identify the linear system with more accurate results than those achieved by using the technique of recursive least squares error identification [36]. The contributions of this paper are: (1) the modelling of WECS with uncertainty parameters is derived by wind speed signal under ARMA model with random process; (2) a QARXNN is applied to model and predict WECS dynamics online with emphasis on the search parameters of the input vector; (3) with minimum variance controller law, a controller signal is derived by QARXNN prediction model using linear and nonlinear parameters with the switching.

4.2 Dynamic Modeling of WECS System

4.2.1 Wind Speed Modeling

Electrical energy generated by wind power plants in any region heavily depends on the characteristics of the wind in that region. There are many ways to model the wind speed as a basis to evaluate the power systems. One way uses the autoregressive moving average model by generating a random signal denoted as $ARMA(p, q)$, in which p refers to the order of autoregressive signals and q to the order of moving average. The ARMA model created for the Swift Current site in Saskatchewan, Canada, based on data regarding the period from 1996 to 2003 appears in the following [54]:

$$\begin{aligned} s(t) = {} & 1.1772s(t-1) + 0.1001s(t-2) - 0.3572s(t-3) \\ & +0.0379s(t-4) + \nu(t) - 0.5030\nu(t-1) \\ & -0.2924\nu(t-2) + 0.1317\nu(t-3) \\ & \nu(t) \in (0, 052476^2). \end{aligned} \tag{4.1}$$

The simulated wind speed at hour t, designated as $V(t)$, can be calculated as follows:

$$V(t) = \mu(t) + \sigma(t)s(t). \tag{4.2}$$

where $\mu(t)$ is the mean observed wind speed per hour, and $\sigma(t)$ is the standard deviation of the observed wind speed per hour.

4.2.2 Dynamic Modeling of WECS

The power captured by a wind turbine is given by

$$P_m = 0.5\rho\pi Cp(\lambda, \beta)R^2V^3 \tag{4.3}$$

where ρ is the air density (typically 1.25 kg/m3), R is radius of blades (in meter), $Cp(\lambda, \beta)$ is the wind-turbine power coefficient, and V is the wind speed (in m/s). The coefficient of $Cp(\lambda, \beta)$ depends on the pitch of the blades β (in degrees) and the tip-speed ratio λ. Tip-speed ratio is defined as the ratio of the linear velocity of the blade tip to the wind speed described as follows:

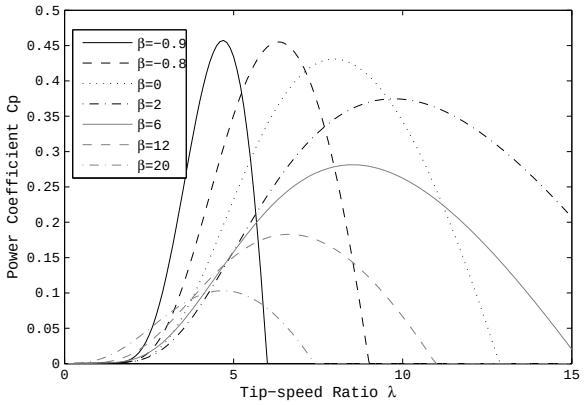


Fig. 4.1. Power coefficient versus tip-speed ratio for various blade pitches β.

$$\lambda = \frac{\omega_t R}{V} \tag{4.4}$$

where ω_t is the wind turbine shaft speed (in rad/s).

The relation of Cp versus λ regarding a three-blade horizontal-axis wind turbine for various blade pitches β is illustrated in Fig. 4.1. The curves have been obtained by plotting (4.5), which is commonly used in wind turbine simulators [55]:

$$Cp(\lambda, \beta) = 0.5176(\frac{116}{\lambda_i} - 0.4\beta - 5)e^{-21/\lambda_i} + 0.0068\lambda \tag{4.5}$$

$$\frac{1}{\lambda_i} = \frac{1}{\lambda + 0.008\beta} - \frac{0.035}{\beta^3 + 1}. \tag{4.6}$$

Fig. 4.2 shows the subsystem's interconnected of WECS that consists of the wind turbine, the drive train, and the generation unit.

The objective of the proposed control is to maximize the power that the turbine extracts, which can be achieved if Cp is maximized. To maximize Cp, λ must be kept constant at its optimum value regardless of wind speed. Fig. 4.3 illustrates the steady-state power-speed characteristics (i.e., solid curves) and the maximum power point curve (i.e., dashed curve) attained for each wind speed at a pitch angle of 0^o. The aerodynamic torque on the wind turbine rotor can be obtained by using the following relationships:

$$T_m = \frac{P_m}{\omega_t} = \frac{\rho \pi Cp(\lambda, \beta) R^3 V^2}{2\lambda}. \tag{4.7}$$

The proposed MPPT technique seeks to retrieve the optimal rotor speed ω_t (i.e., the speed corresponding to the maximum generated power) for any

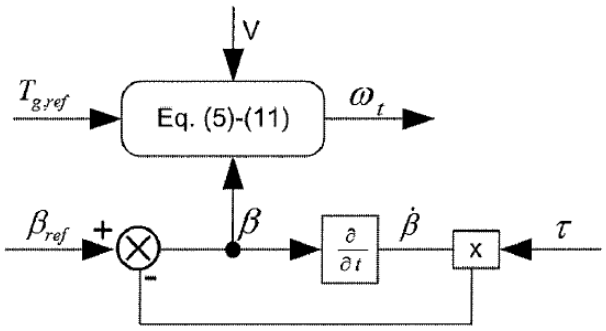


Fig. 4.2. Structural diagram of WECS systems.

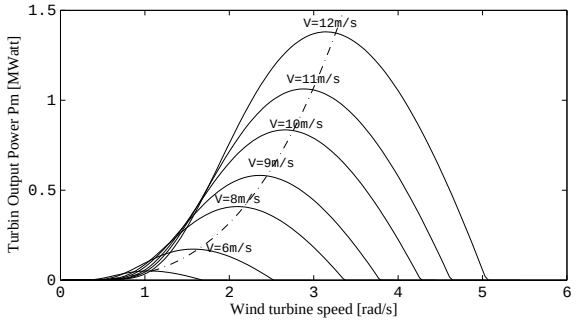


Fig. 4.3. Power-speed characteristics of wind turbines for various wind speeds at a pitch of 0^o.

instantaneous value of wind speed. Note in Fig. 4.2, the external input of the dynamic WECS are the set points of generator torque $T_{g,ref}$, the desired pitch β_{ref}, and wind speed signal V. The outputs of WECS can be measured that presented by the turbine rotor speed ω_t. The desired pitch is the optimum pitch obtained from an aerodynamic turbine characteristic with maximum power coefficient, that can be interpreted as the setting of the pitch position resulting maximum power extracted from wind power. The wind speed signals are fluctuated that can be assumed as a disturbance signal affecting uncertainty parameters of the WECS dynamics. The WECS dynamic can be described as follows:

$$\dot{\theta} = \omega_t - \omega_g \tag{4.8}$$

$$J_g\dot{\omega}_g(t) = K_s\theta + B_s\omega_t - B_s\omega_g + T_g(\omega_g, T_{g,ref}) \tag{4.9}$$

$$J_t\dot{\omega}_t(t) = -K_s\theta - B_s\omega_t + B_s\omega_g + T_m(\beta, V) \tag{4.10}$$

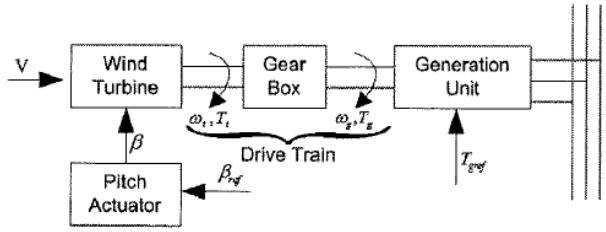


Fig. 4.4. Block diagram of nonlinear dynamic of WECS.

The generator torque T_g is a nonlinear function with the generator speed ω_g and the reference electromagnetic torque $T_{g,ref}$ as a variable. The generator usually operates in the linear region of its torque characteristics, which can be approximated in linear form as

$$T_g = B_g \omega_g - T_{g,ref}. \tag{4.11}$$

The pitch actuator is modelled as a first-order dynamic system with saturation in the amplitude and derivative of the pitch β as [53, 55]

$$\dot{\beta} = \frac{-1}{\tau}\beta + \frac{1}{\tau}\beta_{ref}. \tag{4.12}$$

Fig. 4.4 shows the dynamic of WECS model described in Equations (4.3)-(4.12). The control system acts to control blade pitch position in order to maximize the power extracted from wind, with the reference electromagnetic torque $T_{g,ref}$ set as constant. The system parameters are given as follows [51]:
Turbine and drive train parameters
R=30.30m, K_s=15.66x10$^5 N/m$, Bs=30.29x10$^2 N.ms/rad$,
J_t=83.00x10$^4 kg.m^2$
Generator parameters
B_g=15.99 $N.ms/rad$, J_g=5.9 $kg.m^2$
Pitch actuator
τ=100 ms.

4.3 Control Strategy

To control WECS, the controller is designed in two steps. The first steps involves the identification and prediction of WECS by using QARXNN model, while the second involves deriving and implementing the control law based prediction model. Fig. 4.5 shows an adaptive controller based on QARXNN model. The turbine speed is operated at MPPT point by controlling the pitch blade position with generator torque is assumed to be constant.

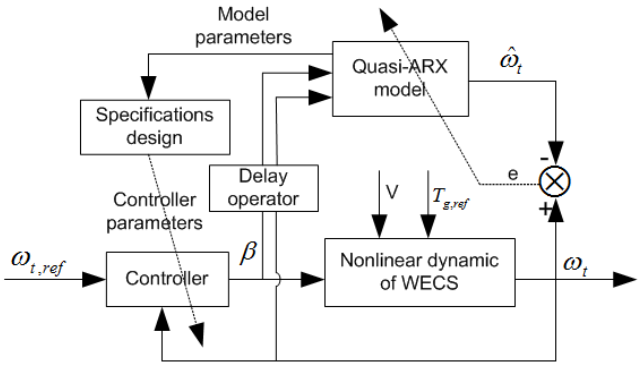


Fig. 4.5. MPPT controller of WECS bases QARXNN prediction model.

4.3.1 System Identification

Through performing Taylor series expansions [13, 16], nonlinear continuous function can be presented as

$$y(t) = y_0 + \phi(t)^T \aleph(\phi(t)) \tag{4.13}$$

where $\aleph(\phi(t)) = [a_{(1,t)} \cdots a_{(n_y,t)}\ b_{(1,t)} \cdots b_{(n_u,t)}]^T$ is a Taylor coefficients, $\phi(t) \in R^{n=n_u+n_y}$ denotes the input vector with elements of $\phi(t) = [-y(t-1) \cdots -y(t-n_y)u(t-1) \cdots u(t-n_u)]^T$, n_u and n_y represents the orders of time delay in input-output data. $\aleph(\phi(t)) \in R^{n=n_u+n_y}$ denotes a kernel function that is used to give coefficients of the input vector. In our main theoretical, the assumption are made as follows:

Assumption 1. The pairs of the input and output of training data are bounded.

Assumption 2. The input and output of nonlinear function $\aleph(\phi(t))$ are bounded.

By performing Taylor series expansion, we develop nonlinear system presented as a linear correlation between the input vector and its coefficients. If the system modelling represents a plant that is a linear system, the coefficients obtained will be constant, other than that, if the system modelling represents a plant that is a nonlinear system, the obtained coefficients will be a function of time [36]. A QARXNN model puts nonlinear function into the coefficients of input vector as follows:

$$\begin{aligned} y(t, \phi(t)) = {} & b_{(1,t)}u(t-1) + \cdots + b_{(n_u,t)}u(t-n_u) \\ & - a_{(1,t)}y(t-1) - \cdots - a_{(n_y,t)}y(t-n_y). \end{aligned} \tag{4.14}$$

Fig. 4.6 illustrates the scheme of system identification and prediction by QARXNN model. The input vector is used as the input for an embedded

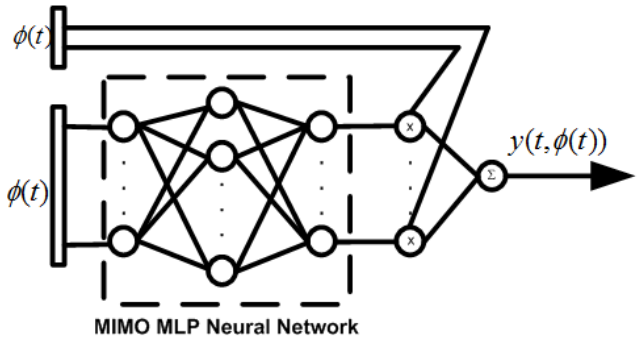


Fig. 4.6. Quasi-ARX neural network with MLP network as embedded systems.

system constructed by MLPNN with three layer neural network. The number input layer, hidden layer and output layer is the same that equals to n. A QARXNN model incorporating neural network can be expressed as

$$y(t,\phi(t)) = \phi(t)^T \aleph(\phi(t)) \tag{4.15}$$

$$\aleph(\phi(t)) = W_2 \Gamma W_1(\phi(t) + B) + \theta, \tag{4.16}$$

where $\Omega = \{W_1, W_2, B, \theta\}$. $W_1 \in R^{mxn}, W_2 \in R^{nxm}, B \in R^{mx1}$ are the weights matrix in the first layer, second layer, and bias vector of hidden nodes. The linear parameters of $\theta \in R^{nx1}$ is searched by performing least square error (LSE) algorithm than is set as a bias vector of output nodes for an embedded system of MLPNN. The symbol of Γ is the diagonal nonlinear operator with identical sigmoidal elements on hidden nodes.

The learning algorithms of QARXNN model executed in two steps are: (1) LSE algorithm is used to update θ and then is set as bias vector for MLPNN; (2) Performing backpropagation (BP) error algorithm for an embedded of MLPNN. By using two algorithms, we introduce two sub-models incorporating linear sub-model and nonlinear sub-model as follows: $z_l(k) = y(t,\phi(t)) - \phi(t)^T[W_2(k)\Gamma W_1(k)(\phi(t)+B(k))]$ and $z_n(k) = y(t,\phi(t)) - \phi(t)^T\theta(k)$, where k denotes a sequence learning number. The learning algorithms for a QARXNN is performed step by step described as follows:

1. set $k = 0$ for initial conditions, $\theta(k) = 0$; and small initial values to $W_1(k)$, $W_2(k)$, and $B(k)$, then set $k = 1$, where k is the learning number.
2. calculate $z_l(k)$, then estimate $\theta(k)$ for by using a least-squares error algorithm.
3. calculate $z_n(k)$, then estimate $W_1(k)$, $W_2(k)$, and $B(k)$. It is realized by using the well-known back-propagation (BP) algorithm.
4. use the (4.16) to update $\aleph(\phi(t))$
5. stop if pre-specified conditions are met, otherwise go to Step 2, and repeat the estimation of $\theta(k)$, and $W_1(k)$, $W_2(k)$, and $B(k)$, set $k = k + 1$.

If model in (4.15) satisfies to model the input-output training data of the system, **Assumption 1.** and **Assumption 2.** are fulfilled, then the output at $(t+d)$ can be predicted. To obtain predicted output, equation (4.15) is regressed at time $(t+d)$ described as [13, 36]

$$\begin{aligned} y(t+d) &= \hat{b}_{(1,t+d)}u(t+d-1) + \cdots + \\ &\hat{b}_{(n_u,t+d)}u(t-n_u+d) - \hat{a}_{(1,t+d)}y(t-1+d) \\ &- \cdots - \hat{a}_{(n_y,t+d)}y(t-n_y+d) \end{aligned} \tag{4.17}$$

where, $\phi(t+d) = [y(t+d-1)\, y(t+d-2) \cdots y(t+d-n_y)\, u(t+d-1)\, u(t+d-2) \cdots u(t+d-n_u)]^T$ is d step ahead of the input vector. $\hat{\aleph}(\phi(t+d))$ and $[\hat{a}_{(1,t+d)} \cdots \hat{a}_{(n_y,t+d)} \hat{b}_{(1,t+d)} \cdots \hat{b}_{(n_u,t+d)}]^T$ are the estimated parameters of nonlinear part by MLPNN and its elements, respectively. For online step ahead prediction, d is equal to one. The estimated coefficient of the input vector is calculated by using an embedded of MLPNN with the next input vector of $d=1$ by $\phi(t+1) = [y(t)\, y(t-1) \cdots y(t+1-n_y)\, u(t)\, u(t-1) \cdots u(t+1-n_u)]^T$. With the next regression vector, the estimated coefficients is calculated by

$$\hat{\aleph}(\phi(t+1)) = W_2 \Gamma W_1(\phi(t+1) + B) + \hat{\theta}. \tag{4.18}$$

By considering the equation of (4.18), we can see that a QARXNN model consists of the estimated linear part parameter of $\hat{\theta}$ and the estimated nonlinear part parameters of $\hat{\aleph}(\phi(t+1))$.

4.3.2 Minimum Variance Based Switching Controller

A QARXNN model is improved to guarantee closed loop stability of control system expressed as

$$y(t, \phi(t)) = \phi(t)^T \hat{\aleph}(\phi(t), \chi(t)) \tag{4.19}$$

$$\hat{\aleph}(\phi(t), \chi(t)) = \chi(t) W_2 \Gamma W_1(\phi(t) + B) + \hat{\theta}. \tag{4.20}$$

where $W_2 \Gamma W_1(\phi(t) + B) + \hat{\theta}$ is the estimated parameters of the nonlinear part, $\hat{\theta}$ denotes the estimated parameters of θ in the linear part that used as a bias vector in the output nodes. Obviously, through introducing the switching function $\chi(t)$, the improved QARXNN model is different from the conventional QARXNN model. When $\chi(t) = 1$, it is a nonlinear prediction model, which can insure the prediction accuracy. And when $\chi(t) = 0$, it is a linear prediction model, which can insure the control stability [20].

The linear part error and nonlinear part error, respectively is defined as follows :

$$e_1(t) = y(t) - \phi(t)^T \hat{\theta}, \tag{4.21}$$

$$\begin{aligned} e_2(t) &= y(t) - \phi(t)^T \hat{\aleph}(\phi(t)) \\ &= y(t) - \phi(t)^T \hat{\theta} \\ &\quad - \phi(t)^T W_2 \Gamma W_1(\phi(t) + B), \end{aligned} \tag{4.22}$$

then θ is updated as:

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \frac{a(t)\phi(t-1)e_1(t)}{1+\phi(t-1)^T\phi(t-1)}. \tag{4.23}$$

$$a(t) = \begin{cases} 1, \text{ if } \|e_1(t)\| > 2\triangle \\ 0, \text{ otherwise} \end{cases} \tag{4.24}$$

Similar to [18, 20, 45], the switching criterion function are described as follows:

$$J_i(t) = \sum_{l=1}^{t} \frac{a(l)(\|e_i(l)\|^2 - 4\triangle^2)}{2(1+\phi(l-1)^T\phi(l-1))} + c\sum_{l=t-N+1}^{t}(1-a(l))\|e_i(l)\|^2, i = 1,2 \tag{4.25}$$

$$\chi(t) = \begin{cases} 1, \text{ if } J_1(t) > J_2(t) \\ 0, \text{ otherwise} \end{cases} \tag{4.26}$$

where i=1 denotes the linear adaptive minimum variance controller, while i=2 denotes the nonlinear adaptive minimum variance controller. The value of $\triangle$ is determined by designer where $\triangle \leq \phi(t)\hat{\aleph}(\phi(t))$, N is a positive integer, and $c \geq 0$ is a predefined as constant. Switching theory and stability analysis of the closed-loop system can be studied more clearly in reference [18–20, 45, 56].

A minimum variance controller is proposed defined as follows:

$$M(t+1) = \left(\frac{1}{2}(y(t+d) - y^*(t+d))^2 + \frac{\lambda}{2}u(t)^2\right) \tag{4.27}$$

where λ is a weight of control input, d is a different operator that equals to 1 for online step ahead prediction. The controller can be obtained by solving

$$\frac{\partial M(t+1)}{\partial u} = 0. \tag{4.28}$$

A QARXNN is used to model WECS online. The controller signal calculated by solving the equation of (4.28) is difficult due to non-linearity and multi-parametric model. Fortunately, a QARXNN model can be simplified by linear correlation between the input vector and its coefficients. The controller is linear to the input variable $u(t)$. Therefore, a controller is derived from simplified QARXNN model [13, 20]. By adapting the input vector, the control law is modified as follows:

$$u(t) = \frac{\hat{b}_1(t)}{\hat{b}_1^2(t)+\lambda}(\hat{b}_1(t) - \hat{b}(q^{-1},\phi(t))q)u(t-1) + y^*(t+1) - \hat{a}(q^{-1},\phi(t))y(t)) \tag{4.29}$$

The controller is designed in two steps: (1) to identify the system using QARXNN model, (2) to design a controller by using the parameter estimation

that has been done in the first step.

Theorem: For a system expressed in (4.19) with using an adaptive minimum variance controller (4.29) and all the input-output signals in the closed-loop system are bounded. Moreover, the tracking error of the system can converge on zero when a properly neural network is determined.

Proof: Firstly, the model error $e_i(t)$ is defined as:

$$\begin{aligned} e_i(t) &= y(t) - \phi(t)^T \hat{\aleph}(\phi(t), \chi(t)) \\ &= y(t) - \phi(t)^T \hat{\theta} \\ &- \chi(t)\phi(t)^T W_2 \Gamma W_1(\phi(t) + B) \end{aligned} \tag{4.30}$$

Then subtracting θ_0 from both sides of (4.23), and gives:

$$\begin{aligned} \tilde{\theta} &= \tilde{\theta}(t-1) \\ &- \frac{a(t)\phi(t-1)(\phi(t-1)^T\tilde{\theta}(t-1) - \delta(t))}{1 + \phi(t-1)^T\phi(t-1)} \end{aligned} \tag{4.31}$$

where $\tilde{\theta} = \hat{\theta}(t) - \theta_0$ and $\delta(t) = y(t) - \phi(t)^T\theta\hat{(t)}$.

consider a function as follows:

$$V(t) = \left\|\tilde{\theta}(t)\right\|^2. \tag{4.32}$$

Then, nothing that $a(t)$=0 or 1, and combined with (4.23) and (4.24), we can get:

$$\begin{aligned} V(t) &= V(t-1) - \frac{2a(t)(e_i(t) - \delta(t))e_i(t)}{1 + \phi(t-1)^T\phi(t-1)} \\ &+ \frac{a(t)^2\phi(t-1)^T\phi(t-1)e_i(t)^2}{(1 + \phi(t-1)^T\phi(t-1))^2} \\ &\leq V(t-1) + \frac{a(t)(2e_i(t)\delta(t))}{1 + \phi(t-1)^T\phi(t-1)} \\ &- \frac{a(t)e_i(t)^2}{1 + \phi(t-1)^T\phi(t-1)} \end{aligned} \tag{4.33}$$

from $2ab \leq Ca^2 + b^2/C, \forall C$, the following inequality holds:

$$\begin{aligned} V(t) &\leq V(t-1) + \frac{a(t)(e_i(t)^2/2 + 2\delta(t)^2)}{1 + \phi(t-1)^T\phi(t-1)} \\ &- \frac{a(t)e_i(t)^2}{1 + \phi(t-1)^T\phi(t-1)} \\ &\leq V(t-1) + \frac{2a(t)\triangle^2}{1 + \phi(t-1)^T\phi(t-1)} \\ &- \frac{1}{2}\frac{a(t)e_i(t)^2}{1 + \phi(t-1)^T\phi(t-1)} \end{aligned} \tag{4.34}$$

$V(t)$ is nonincreasing sequence bounded by zero. Moreover,

$$\lim_{N\to\infty}\sum_{t=1}^{N}\frac{a(t)(e_i(t)^2-4\triangle^2)}{2(1+\phi(t-1)^T\phi(t-1))}<\infty, \tag{4.35}$$

and

$$\lim_{N\to\infty}\frac{a(t)(e_i(t)^2-4\triangle^2)}{2(1+\phi(t-1)^T\phi(t-1))}\to 0. \tag{4.36}$$

The stability of the closed system via switching technique for the adaptive minimum variance controller can be described as follows: $J_1(t)$ is always bounded by (4.24) and (4.35). $J_2(t)$ has two cases:

1. $J_2(t)$ is bounded. so the model error $e(t)$ is bounded and satisfies (4.36).
2. $J_2(t)$ is unbounded. Since $J_1(t)$ is bounded. So there exists a constant t_0 such that $\chi(t)=0, \forall t>t_0$. The model also has bounded error $e(t)$.

From above inequalities, the input and output of the closed-loop switching control system are bounded. The linear control system is always bounded. If a proper nonlinear model is chosen and the accurate parameters is adjusted, the nonlinear control error $e_2(t)$ can converge on zero. The model only with linear parameters has to work until the use of nonlinear parameters is not disturb the stability of closed loop system. Therefore, the controller with using linear parameters $\hat{\theta}$ will work all the time, but the nonlinear parameters $\hat{\aleph}(\phi(t))$ will work under the switching sequence.

In Chapter 3, we have discuss about switching control performed based on Lyapunov stability theory. A new switching mechanism is proposed based on the state of dynamic tracking error so that more information will be provided, not only error but also one up to p-th differential error will be available as the switching variable. The switching mechanism is derived based on Lyapunov stability theorem by utilizing the state parameter of dynamic tracking error obtained from prediction model. Therefore, the switching mechanism become more effective and efficient. Moreover, the proposed switching formula can use the parameter of prediction model presented by SDPE as variables of switching condition criterion. In order to guarantee closed loop control stability Lyapunov based switching control is used presented in (4.35) to track MPPT of WECS. In this Chapter we show the performance of the proposed switching control compared to the previous one that is based on convergence index based switching law with minimum variance controller.

4.4 Simulation and Results

The proposed MPPT control strategy is applied to arrange a pitch of blade β in order to track the angular velocities of a turbine operating at MPPT point.

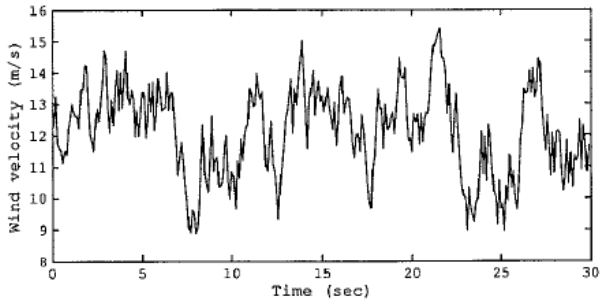


Fig. 4.7. Wind speed.

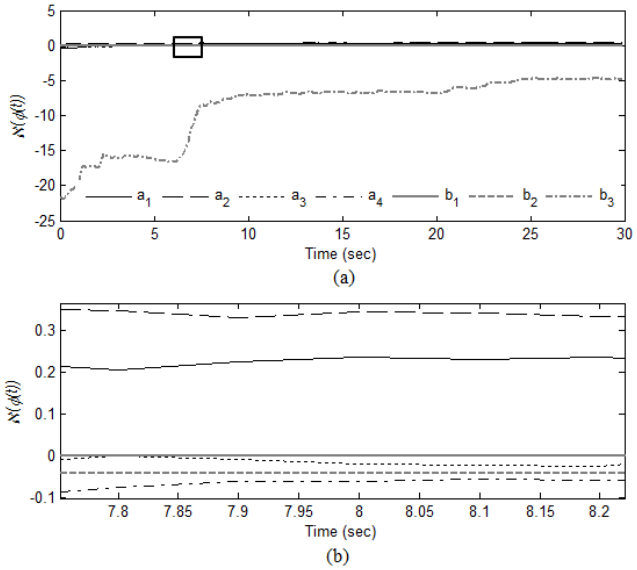


Fig. 4.8. The parameters of the input vector $\hat{\aleph}(\phi(t))$.

Wind speed is generated by an ARMA model shown by Fig. 4.7, the mean observed wind speed of $\mu(t) = 12$ m/s and the standard deviation (SD) of the observed wind speed of $\sigma(t) = 1.5$. The simulation results are shown in detail in Fig. 4.7 - Fig. 4.13. In order to obtain the maximum output of power from a wind turbine generator system, it is necessary to drive the wind turbine at an optimal rotor speed for a particular wind speed.

An embedded system is constructed by MLPNN with three layer neural network. The input vector of $\phi(t)$ is selected by $\phi(t) = [y(t-1)\, y(t-2)\, y(t-3)\, y(t-4)\, u(t-1)\, u(t-2)\, u(t-3)]^T$ with $n = 7$ equal to the sum of $n_u = 3$ and $n_y = 4$. The number of input nodes, hidden nodes, and output nodes is also the same as n. The parameter of the switching criterion c=1.2 and

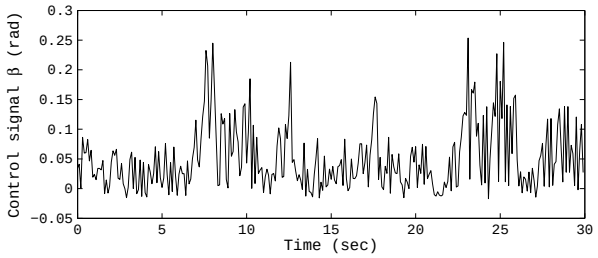


Fig. 4.9. Control signal.

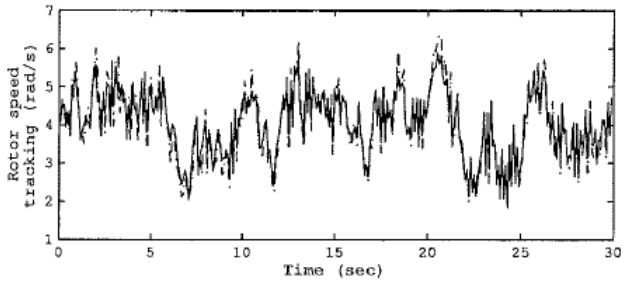


Fig. 4.10. Trajectory of ω_t of minimum variance controller with switching based quasi-ARX model.

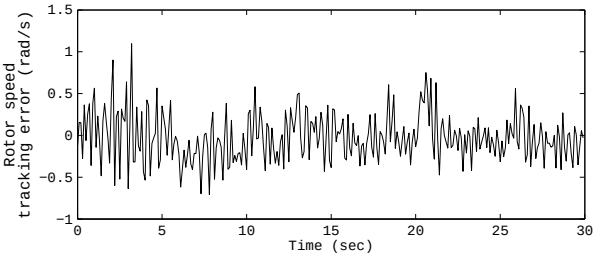


Fig. 4.11. Tracking error of turbine angular velocity.

N=3. By using TS fuzzy model with parameter uncertainties [51], the parameter (stiffness, damping, and moments of inertia) of the WECS is considered as uncertainty parameter. It is presented by nominal parameter (fixed) and uncertainty parameter (noise). In this paper, the WECS system is modelled under QAXNN model. The parameter consist of two parts: (1) nonlinear parameter $W_2 \Gamma W_1(\phi(t) + B)$ which can be regarded as uncertainty parameter executed by MLPNN and (2) linear parameter θ. Linear relationship between the parameter and the input vector, is to make easy to derive control law. Switching mechanism is used to maintain system stability. By performing

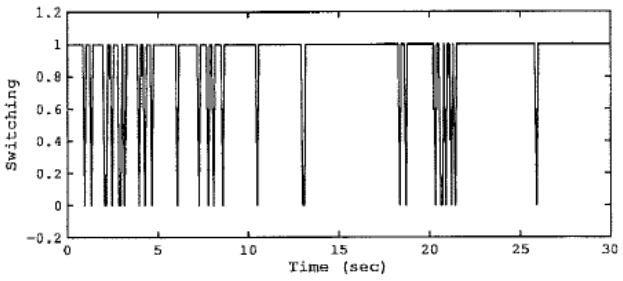


Fig. 4.12. Switching sequance.

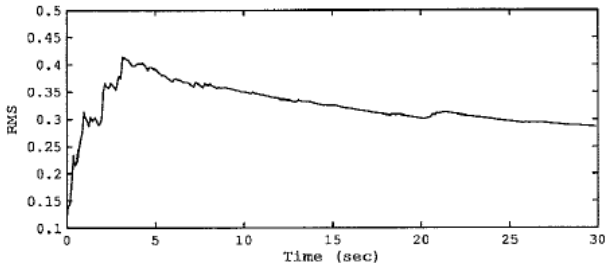


Fig. 4.13. RMS error versus time.

system identification, the parameter of input vector $\aleph(\phi(t))$ can be estimated. As we can see that the WECS is nonlinear system influenced by wind speed fluctuation, thus rendering the parameters uncertain or time function shown by Fig. 4.8. By using a technique of minimum variance controller, control signal is calculated by using $\aleph(\phi(t))$ set as controller parameter with switching law. To make more contrast, the snapshot of the part marked in Fig. 4.8a is enlarged shown by Fig. 4.8b.

Control signal shown by Fig. 4.9 is fed to the WECS system to track wind turbine rotor speed where the result is shown by Fig. 4.10. The dot-dash line denotes speed references ω_t at MPPT operating point and solid line denotes the output of the proposed method, respectively. The tracking error of turbine rotor speed is shown in Fig. 4.11. Switching function between nonlinear and linear parts to ensure the closed-loop stability and to improve the control accuracy is shown by Fig. 4.12. The performance of the proposed controller is also measured by rooted mean squared (RMS) error index versus time shown in Fig. 4.13.

$$RMS = \sqrt{\frac{\sum_{t=1}^{N}(y^*(t) - y(t))^2}{t}} \tag{4.37}$$

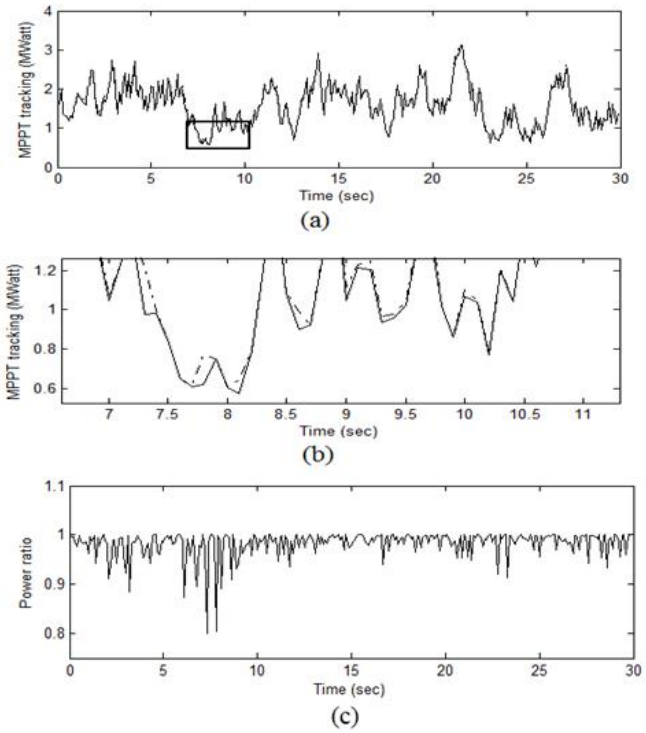


Fig. 4.14. Tracking MPPT power compared to aerodynamic power. (a) MPPT power tracking. (b) the snapshot of the part marked in (a) enlarged. (c) the ratio between the tracked MPPT power to aerodynamic power

where $y^*(t)$ denotes the reference signal and $y(t)$ denotes the output of controlled plant. Fig. 4.14 illustrate the WECS response at the MPPT operating point for the initial condition $t = 0s$,$V = 12.48m/s$, the power of MPPT aerodynamic tracking $1.45MW$, $\beta = 0^0$, angular velocity $\omega_t = 4.12rad/s$. When wind speed change either to decrease or increase, the rotor speed of turbine also should be change in order to keep maximum power of WECS by controlling blade pitch ratio β.

If the rotor speed operates at speed references ω_t then the maximum power coefficient can be achieved. Thus the turbine generator output is maximum. The result of MPPT power tracking is shown by Fig. 4.14. The tracked MPPT power using the proposed controller is compared to aerodynamic power shown by Fig. 4.14a, dot-dashed line shows the aerodynamic power and dashed line shows the tracked MPPT power. To make contrast, the snapshot part marked in Fig. 4.14a is enlarged shown by Fig. 4.14b. The accuracy of the tracked MPPT power is shown by Fig. 4.14c presented by a ratio between the tracked MPPT power to aerodynamic power. Ratio of 1 indicates an accuracy of

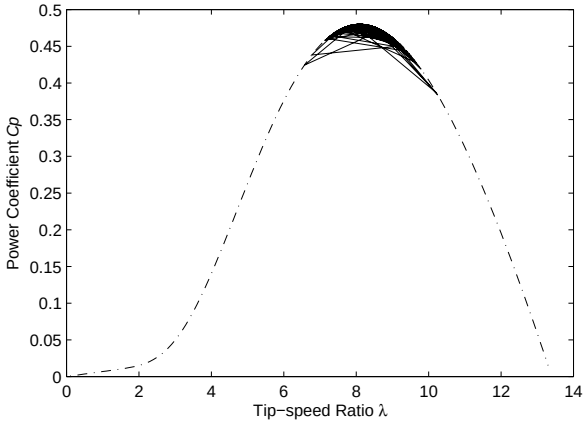


Fig. 4.15. Tracking Cp compared to the characteristic of turbine.

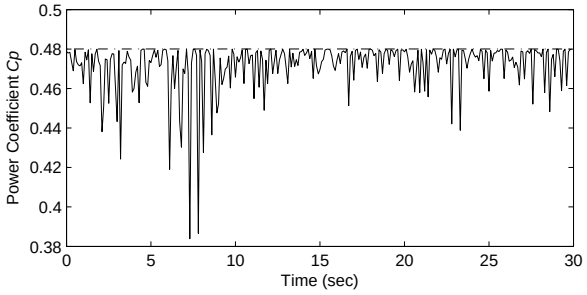


Fig. 4.16. Tracking Cp versus time.

100 %. The performance of tracking control also can be measured, shown by Fig. 4.15 and Fig. 4.16. Turbine peak performance is achieved with $Cp = 0.48$. Dot-dashed line shows Cp characteristics of the turbine and dashed-line shows the result of the tracked Cp. These can also be served by Cp versus time in Fig. 4.16.

Under reference [51], the resolution pitch actuator is 0.1 seconds. Therefore, a sampling time of 0.1 seconds is used for the tracking MPPT control simulation, so it seems to high-speed control. However, to reduce the high speed control, the sampling time can be changed and adapted to the real application and the characteristic of the existing turbine components.

4.5 Conclusion

This paper presents an adaptive controller using prediction model of QARXNN. Based on the result of simulation, a minimum variance controller based on

QARXNN prediction model is effective to track MPPT of the WECS. The proposed method is executed step by step as follows: (1) wind speed dynamic model is adopted bases autoregressive moving average model by generating a random signal; (2) the principles of dynamic modelling of WECS is derived with given parameters where the maximum energy that can be extracted of wind power is influenced by wind speed and pitch of the blades; (3) The dynamic of WECS is simulated and identified online using quasi-ARX neural network model. The next input regression vector is the input for an embedded system of MLPNN to estimate parameters that is used directly as controller parameters. The controller work under switching law to guarantee closed-loop stability. Finally, the control performance has been confirmed by a simulation and experimental results. The main contributions of this study are: 1) the successful development of nonlinear dynamics of WECS modelling bases wind speed dynamic of ARMA model with generating a random signal; 2) the successful application of the QARXNN prediction model to predict WECS online; 3) the successful application of switching controller based QARXNN to track MPPT of the WECS.

5

The Estimated Parameters using Two Steps Identification under QARXNN model

5.1 Introduction

System identification has axiomatic importance in modern life. In control system engineering, system identification methods give appropriate models for characterizing dynamic behavior of a given system [57]. The one of identification scheme is linear time invariant system (LTI) that can be presented by autoregressive (AR), moving average (MA), and autoregressive moving average (ARMA). The LTI models have been used in various applied problems involving system identification, time-series analysis, spectral analysis, or deconvolution [58–60]. The LTI modelling also has been successfully applied to many applications in several areas, such as signal processing [61–63], biology and biomedical engineering [64–66], image processing [67–69], and control engineering [70, 71].

In the area of systems and control, the control mostly is of feedback type and is designed to optimize some performance index [72]. The one of performance index is convergence speed of parameter estimation. It is a significance problem instead of function approximation. The estimated parameter using recursive least square (RLS) algorithm is a function over time. The convergence analysis is solved to deal with a differential equation that contains all necessary information about the asymptotic behavior [73]. It is well known that in the deterministic case, the RLS estimates always converge and that the convergence is to the true parameters if the data covariance matrix grows to infinity [74]. Several approaches have been developed in order to speed up the convergence of the estimated parameters and its accuracy such as dichotomous coordinate descent recursive least square (DCD-RLS) algorithm [75] and the refined instrumental variable method for continuous time systems (RIVC) [76].

In this paper, a dual Mapping Hierarchically under Quasi Linear-ARX Model is proposed to estimate the linear systems. First, the system is identified under least square error algorithm (LSE). The estimated parameters of LSE is used as bias vector for the output nodes of neural network (NN).

Second, refining the estimated parameters based on the residual error of LSE algorithm. Depending the dynamic behavior of the system, the estimated parameters cannot directly converge to the true parameters. Therefore, the convergence characteristic of the estimated parameters is assumed as a nonlinear function performed by NN. To make fast convergence, NN is trained by reducing the error of the LSE algorithm hierarchically. Thus, the convergence speed of the estimated parameters can be improved.

A quasi linear-ARX neural network (QARXNN) is composed hierarchically using linear and nonlinear subsystem modelling. It has two linear and nonlinear sub-models to parameterize the regression vector [13, 16, 28]. Dual mapping performs the coefficient of the regression vector. First, the system is identified using the least square error (LSE) algorithm utilizing the linear sub-model. From the LSE algorithm, we have the estimated parameters of the linear subsystem modeling. However, the estimated parameters of LSE cannot converge to the true parameters. The convergence characteristic of the LSE is to the true parameters when the information vector goes to infinity. Second, nonlinear sub-model is performed to make fast convergence of the estimated parameters with a few number of information vector. The linear parameter estimation (LPE) is set as the bias vector of output nodes for the nonlinear sub-model performed by NN. Thus, the error of the linear sub-model can be reduced by performing the nonlinear sub-model. Finally, experiments and numerical simulations reveal that the proposed method gives satisfactory results to improve the convergence speed of the estimated parameters.

5.2 Problem Description

Assume that the unknown nth-order linear time-invariant discrete-time system in Fig. 5.1 can be described by:

$$A(z^{-1})y(k) = B(z^{-1})u(k) + \omega(k) \tag{5.1}$$

where

$$A(z^{-1}) = 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_n z^{-n}$$
$$B(z^{-1}) = b_1 z^{-1} + b_2 z^{-2} + \cdots + b_m z^{-m}$$

and z^{-l} is the unit delay operator (i.e. $z^{-1}y(k) = y(k-l)$). $u(k)$ and $y(k)$ denote the (scalar) system input and observed noisy output, respectively. $\omega(k)$ is a stochastic white noise with zero mean and variance σ_ω^2. The model is with zero initial conditions ($y(k) = 0$, $u(k) = 0$, for $k < 0$). The structure parameters of the identified system, n and m are assumed to be known. The input $u(k)$ and the disturbance $\omega(k)$ are mutually uncorrelated statistically. The independence between $u(k)$ and $\omega(k)$ means that the system is assumed to operate in open loop.

In matrix equation system (5.1) can be described by

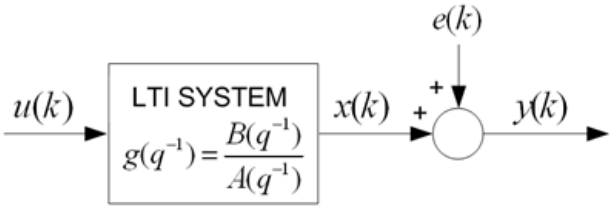


Fig. 5.1. LTI systems with coloured noises

$$y(k) = \phi^T(k)\theta + \omega(k) \tag{5.2}$$

where

$$\theta = [a_1, a_2, ..., a_n, b_1, b_2, ..., b_m]^T \in R^{n+m}$$
$$\phi(k) = [-y(k-1)... - y(k-n)\ u(k-1)...u(k-m)]^T$$

θ and $\phi(k)$ are the parameter vector and the regression vector composed of delayed input-output data, respectively.
Assumption 1. It is assumed that A is stable, i.e $A(z^{-1}) = 0$ implies $\mid z \mid < 1$.
Assumption 2. The pairs of the input $u(k)$ and output $y(k)$ of training data are bounded.

The LS estimation criterion is defined as

$$J_N(\theta) = \sum_{k=1}^{N} (y(k) - \phi^T(k)\theta)^2. \tag{5.3}$$

To analyse the asymptotic property of the LS estimate, which follows from the ergodic theorem [77] that

$$\lim_{N\to\infty} \frac{1}{N}\phi^T(k)\phi(k) = \lim_{N\to\infty} \hat{R}_{\phi\phi}(N)$$
$$= R_{\phi\phi} = E[\phi^T(k)\phi(k)] \tag{5.4}$$
$$\lim_{N\to\infty} \frac{1}{N}\phi^T(k)y(k) = \lim_{N\to\infty} \hat{R}_{\phi y}(N)$$
$$= R_{\phi y} = E[\phi^T(k)y(k)] \tag{5.5}$$
$$\lim_{N\to\infty} \frac{1}{N}\phi^T(k)\omega(k) = \lim_{N\to\infty} \hat{R}_{\phi\omega}(N)$$
$$= R_{\phi\omega} = E[\phi^T(k)\omega(k)]. \tag{5.6}$$

$\hat{\theta}_{LS}(N)$ is the estimated parameter, which can be described as

$$\hat{\theta}_{LS}(N) = \hat{R}^{-1}_{\phi\phi}(N)\hat{R}_{\phi y}(N) \tag{5.7}$$

When N tends to infinity, from (5.2) and (5.7), it can be stated as [77]

$$\lim_{N\to\infty} \hat{\theta}_{LS}(N) = \theta + R_{\phi\phi}^{-1} R_{\phi\omega}. \tag{5.8}$$

Analytical minimisation of (5.3) leads to the LS estimate of θ [77] :

$$\hat{\theta}_{LS}(N) = [\sum_{k=1}^{N} \phi^T(k)\phi(k)]^{-1} [\sum_{k=1}^{N} \phi^T(k) y(k)] \tag{5.9}$$

If $\omega(k)$ is white noise, it is easy to deduce that the disturbance $\omega(k)$ and the vector of lagged input-output data $\phi(k)$ are mutually uncorrelated, and hence $R_{\phi\omega} = 0$. It follows from (5.8) that $\theta_{LS}(N) \to \theta$ with probability one as $N \to \infty$ [78]. Relation (5.9) is often denoted as the least squares estimator. It is convenient to postulate that $\sum_{k=1}^{N} \phi^T(k)\phi(k)$ is a nonsingular matrix and has an inverse.

Equation (5.2) represents for $k = 1, 2, \cdots, N$ a set of linear algebraic equations which may be written in the vector-matrix form as

$$y = \phi^T \theta + \omega \tag{5.10}$$

where

$$\begin{aligned} y &= [y(1), y(2), \cdots, y(N)]^T \\ \omega &= [\omega(1), \omega(2), \cdots, \omega(N)]^T \\ \Phi &= [\phi(1), \phi(2), \cdots, \phi(N)]^T \end{aligned}$$

The elements of Φ are described such as
$\phi(1) = [-y(k-1), \cdots, -y(k-n), u(k-1), \cdots, u(k-m)]^T$
$\phi(N) = [-y(N-1), \cdots, -y(N-n), u(N-1), \cdots, u(N-m)]^T$. The mapping of parameters and data in vectors and matrices in (5.2) and (5.10) respectively need not be necessarily as indicated above. For statistical and probabilistic considerations the number of observations N needs to be much larger than the number $2m + 2n$ of parameters to be estimated [79].

Theorem 1. The quadratic cost function (5.3) attains an absolute minimum if and only if $\theta = \hat{\theta}_{LS}(N)$.

Theorem 2. The estimate of parameters of the regression model (5.2) in the sense of least squares is unbiased if the mean values of the components of the noise-vector ω are equal to zero and if the matrix Φ, and the noise-vector ω are mutually independent.

Theorem 3. If the measurement matrix Φ, and the noise-vector ω are mutually independent, $E[\omega] = 0$, $E[\omega^T \omega] = \sigma_\omega^2 I$, where I is the identity matrix, then

$$E[(\hat{\theta}_{LS}(N) - \theta)^T (\hat{\theta}_{LS}(N) - \theta)] = \sigma_\omega^2 (\Phi^T \Phi)^{-1} \tag{5.11}$$

where $E[\omega]$ is the information concerning the noise, $E[\omega^T \omega]$ is the the covariance matrix of the noise, particularly

$$E[\omega^T\omega] = \begin{bmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \sigma_N^2 \end{bmatrix}$$

In many applications it is important to estimate the parameter vector θ recursively (or on-line or sequentially) as more information becomes available. The basic algorithms for this are RLS algorithm. The estimation of $\hat{\theta}(k)$ recursively are described as [80]

$$\hat{\theta}_{LS}(k) = \hat{\theta}_{LS}(k-1) - P(k)\phi(k) \\ (y(k) - \phi^T(k)\hat{\theta}_{LS}(k-1)) \tag{5.12}$$

$$P^{-1}(k) = P^{-1}(k-1) - \phi(k)\phi^T(k) \tag{5.13}$$

or

$$P(k) = P(k-1) - \frac{P(k-1)\phi(k)\phi^T(k)P(k-1)}{1 + \phi^T(k)P(k-1)\phi(k)}$$

where $P(0) = p_0 I$, $P(k) \in R^{n\mathbf{x}n}$ is the covariance matrix, p_0 is a large positive number.

5.3 Deep Searching Parameter Estimation

Consider a single-input single-output (SISO) black box time invariant whose the input output relation described by:

$$y(k) = g(\phi(k)) + \omega(k) \tag{5.14}$$

where $g(\cdot)$ is a unknown function. By performing Taylor expansion series [13, 14, 16, 36], system in (5.9) can be presented as

$$y(k) = y_0 + \phi^T(k)\theta(\phi(k)) + \omega(k). \tag{5.15}$$

We can see that (5.10) is similar in form to (5.2). Therefore, it is possible to identify LTI system by two steps identification process shown in Fig. 5.2.

By using LS algorithm, θ is determined by minimizing a criterion based on the equation error in surface sub-model describe by

$$J_N(\theta) = \sum_{k=1}^{N} (y(k) - \phi^T(k)\theta)^2. \tag{5.16}$$

Analytical minimisation of (5.11) leads to the least square (LS) estimate of θ [78]:

$$\theta_{LS}(N) = [\sum_{k=1}^{N} \phi^T(k)\phi(k)]^{-1}[\sum_{k=1}^{N} \phi^T(k)y(k)] \tag{5.17}$$

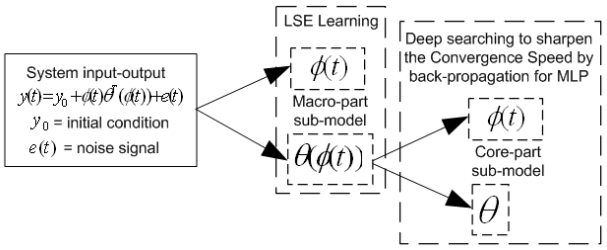


Fig. 5.2. Deeper searching by two steps identification process.

where N is time moving window of regression vector. The output of surface sub-model can be calculated by

$$y_{LS}(k) = \phi^T(k)\theta_{LS}(N). \tag{5.18}$$

We define residual error of LS estimation described as

$$\begin{aligned} e_{LS}(k) &= y(k) - y_{LS}(k) \qquad (5.19)\\ &= \phi^T(k)\theta - \phi^T(k)\theta_{LS}(N)\\ &= \phi^T(k)(\theta - \theta_{LS}(N))\\ &= \phi^T(k)\Delta\theta \qquad (5.20) \end{aligned}$$

where $\Delta\theta$ are performed by MIMO in bottom sub-model. Assume that there exists optimal solution of $\hat{\theta}$ such that satisfies

$$y(k) = \phi^T(k)\hat{\theta} \tag{5.21}$$

The adaptation parameter estimate is defined as

$$\hat{\theta}(k) = \theta_{LS}(N) + \Delta\theta \tag{5.22}$$

Remark 1: In the Coupled LS-BP algorithm, the convergence of LS algorithms in surface sub-model are improved by backpropagation algorithm in MIMO bottom sub-model. The residual error e_{LS} are utilized to refine estimated parameter through sharpen learning by performing iteration in dual process identification. The more accurate the LS performance and the longer the memory span over where the LMS algorithm remembers past data will be. However, the convergence rate of the algorithm is slow. Our motivation has been to achieve faster convergence and limited memory span by time moving window without sacrificing the simplicity of LS algorithm.

In coupled BP-LS algorithm, the output of network with dual sub-model can rewritten as

$$y(k) = \phi^T(k)\theta_{LS}(N) + \phi^T(k)\delta\theta. \tag{5.23}$$

The bottom sub-model is performed by the MIMO feedforward neural network model expressed as

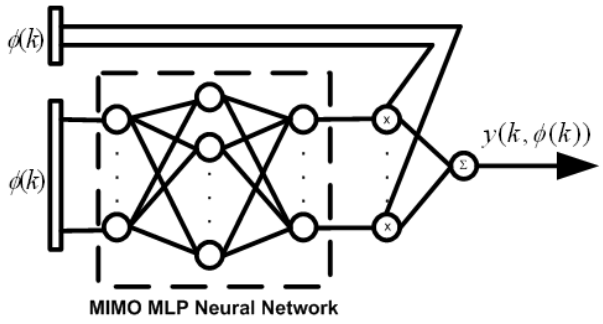


Fig. 5.3. Quasi-ARX neural network model.

$$\Delta\theta(\phi(k)) = W_1(\phi(k)) + B \tag{5.24}$$

where, $\Delta\theta = \{W_1, B\}$, $W_1 \in R^{n+m \mathbf{x} n+m}$ and $B \in R^{n+m}$ are the weight matrix at the first to output layer and the bias vector of output nodes, respectively. The bottom sub-model incorporating neural network of quasi-ARX model is shown in Fig. 5.3.

5.4 Learning Algorithm for Quasi-ARX Neural Network

The learning algorithm for Quasi-ARX model is performed by the back propagation error algorithm for the core-part sub-model, and LSE algorithm for the macro-part sub-model. Let we introduce two sub-models $z_l(k) = y(k, \phi(k)) - \phi(k)\Delta\theta^T(\phi(k))$, and $z_n(k) = y(k, \phi(k)) - \phi(k)\varrho^T$ to become learning guide. The learning output guidance to train sub-models are expressed as,

$$SM1 \quad z_l(k) = \phi(k)\varrho^T(k). \tag{5.25}$$

$$SM2 \quad z_n(k) = \phi(k)\Delta\theta^T(\phi(k)). \tag{5.26}$$

The step of learning algorithm of Quasi-ARX neural network is described by,

1. set $\varrho = 0$; and small initial values to W_1, W_2, and B, set $i = 1$, where i is the learning number.
2. calculate $z_l(k)$, then estimate ϱ for sub-model $SM1$ by using a least-squares error algorithm.
3. calculate $z_n(k)$, then estimate W_1, W_2, and B for sub-model $SM2$ This is realized by using the well-known back-propagation (BP) algorithm.
4. use the (**??**) to update $\theta(\phi(k))$
5. stop if pre-specified conditions are met, otherwise go to Step 2, and repeat the estimation of ϱ, and W_1, W_2, and B, set $i = i + 1$.

5.5 Recursive least-squares Algorithm

Least-squares algorithm is very well known algorithm applied to estimate the parameter estimation of the LTI system. Many researchers have made improvement, modify and development of RLS algorithm to increase the accuracy of the estimated parameter estimation [73**?** , 74]. The RLS estimation techniques are the fundamental technique in adaptive signal processing applications. The equation of (5.10) is the problem that will be solved by Quasi-ARX neural network, then the results are compared to RLS algorithm.

The Algorithm of RLS is performed by minimizing cost function stated as,

$$J(\theta) = \sum_{k=1}^{k_n} (y(k) - \phi(k)\theta^T(k)) \tag{5.27}$$

where k_n is the last sampling number. Assume that the matrix $\phi(k)$ has full rank, so $\phi(k)\phi^T(k)$ is nonsingular for all k. The estimation of $\hat{\theta}(k)$ recursively are described as,

$$\hat{\theta}(k) = \hat{\theta}(k-1) - K(k)(y(k) - \phi(k)\hat{\theta}^T(k-1)). \tag{5.28}$$

$$K(k) = \frac{P(k-1)\phi^T(k)}{1 + \phi(k)P(k-1)\phi^T(k)}. \tag{5.29}$$

$$P(k) = [1 - K(k)\phi^T(k)]P(k-1). \tag{5.30}$$

5.6 Experimental Studies

The Quasi-ARX neural network model is applied to identification Linear Time Invariant (LTI) system. The Pseudo Random Binary Sequence Signal (PRBS) is as input. The system is added with zero mean noise. The system example which is applied to measure performance identification is switched mode power converters (SMPC) in [75]. The SMPC discrete transfer function is stated as,

$$G(q^{-1}) = \frac{0.226q^{-1} + 0.1118q^{-2}}{1 - 1.914q^{-1} + 0.949q^{-2}}. \tag{5.31}$$

The output of the system has a ripple caused by signal perturbation approximately 10% or source to noise ratio (SNR) 20 dB of the system output. The SNR gaussian noise is expressed as,

$$SNR = \texttt{10log}(\frac{\sum_{k=1}^{N} x(k)^2}{\sum_{k=1}^{N} e(k)^2})dB. \tag{5.32}$$

The core-part sub-model is performed by MLP neural network. The number of input node n is the sum of n_u=2 and n_y=2, and the information vector is as the

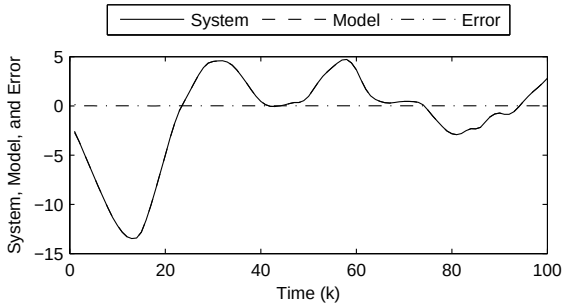


Fig. 5.4. System Output, Model Output and Error of the Quasi-ARX for step ahead prediction

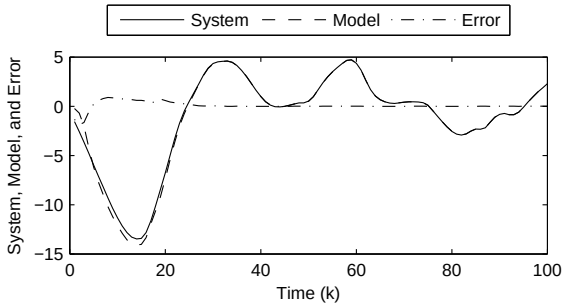


Fig. 5.5. System Output, Model Output and Error the RLS Algorithm for step ahead prediction

input , the number of hidden node and output node is the same as input. The result of system identification which represent the accuracy of input-output of the system by Quasi-ARX neural network is shown in Fig. 5.4 and Fig. 5.6. The accuracy of input-output of the system by the RLS algorithm is shown in Fig. 5.5 and Fig. 5.6. The MSE error over time is shown in Fig. 5.6, the red line represent the MSE error by RLS algorithm and blue line by Quasi-ARX model.

The performances of system identification are measured by Mean Squares Error (MSE)Index to show the accuracy of system input-output.

$$MSE = \frac{\sum_{k=1}^{N}(y_p(k) - y(k))^2}{N}. \tag{5.33}$$

The accuracies of the estimated parameter are measured by Root Mean squares of Parameter (RMSP) stated as,

$$RMSP = \sqrt{\frac{\sum_{k=1}^{N}\sum_{r=1}^{n_u+n_y}(\theta(r) - \hat{\theta}(r))^2}{N(n_u + n_y)}}. \tag{5.34}$$

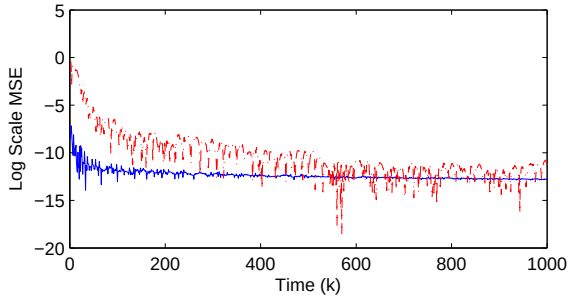


Fig. 5.6. The MSE Index Performance for step ahead prediction

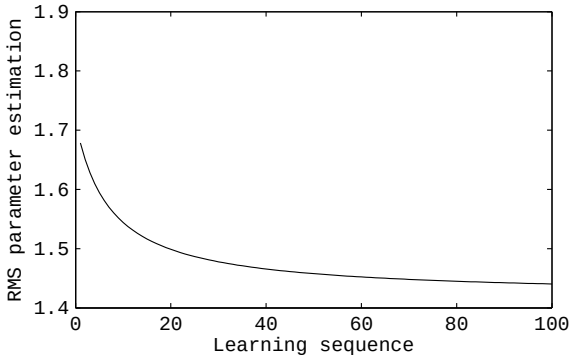


Fig. 5.7. The estimated parameter by using Quasi-ARX neural network

The parameter estimation and its accuracy are shown in Fig. 5.7 and Fig. 5.9 by using Quasi-ARX neural network, and are shown in Fig. 5.8 and Fig. 5.9 by using RLS algorithm. The solid line is the true value, and the dashed line is the estimated value. The blue line, red line, black line and green line are the parameter estimation of a_1, a_2, b_1 and b_2. The RMSP error over time is shown in Fig. 5.9, the red line represent the MSE error by RLS algorithm and blue line by Quasi-ARX model.

The performance of the estimated parameter (EP) by Quasi-ARX is shown in Table 5.1, and the EP by RLS algorithm is shown in Table 5.2. The index of ARMSP is 0.00247 by Quasi-ARX, and 0.329 by RLS algorithm.

5.7 Result and Discussion

In this system identification, we introduce to apply Quasi-ARX model to estimate parameter for the LTI system. Therefore, we pay much attention in the accuracy of the parameter estimation instead of the accuracy of the system input-output. The fast convergence to find the estimated parameter

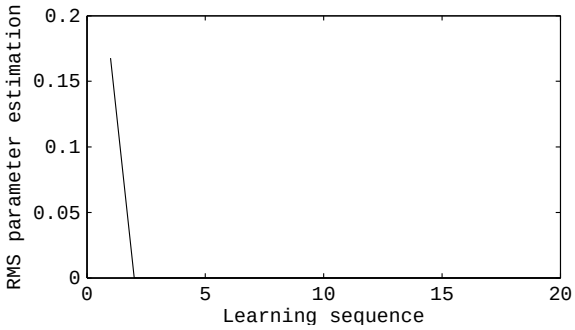


Fig. 5.8. The estimated parameter by using RLS algorithm

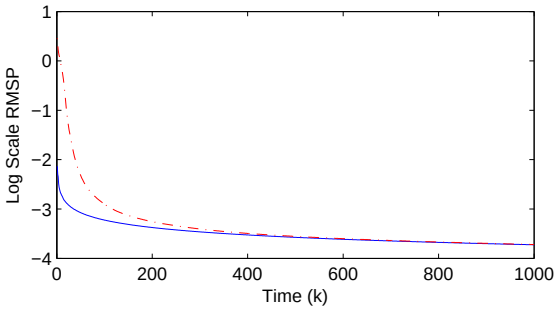


Fig. 5.9. The RMSP Index Performance for step ahead prediction

Table 5.1. Estimated parameter are tracked by using the Quasi-ARX neural network model.

	a_1	a_2	b_1	b_2
True	-1.914	0.949	0.226	0.1118
Mean	-1.912	0.946	0.231	0.1141
MSE	3.57E-06	2.07E-05	9.61E-09	2.19E-07
RMS	1.89E-03	4.55E-03	9.80E-05	4.68E-04

are to be sharpened by the core-part sub-model of Quasi-ARX neural network. We can see the accuracy of the model, which is shown in Fig. 5.4, Fig. 5.5 and Fig. 5.6. It indicates that the Quasi-ARX neural network model is more accurate model compared to the RLS model algorithms.

The performances in searching of the estimated parameter estimation are shown in Fig. 5.7,Fig. 5.8, Fig. 5.9, Table 5.1, and Table 5.2. We can see that the Quasi-ARX neural network is the more accurate model to estimate the parameter estimation of the LTI system. The average of RMS error of the parameter estimation (RMSP) in one hundred sampling is 0.00247 identified by the Quasi-ARX model, and is 0.329 identified by RLS algorithm. The

Table 5.2. Estimated parameter are tracked by using the RLS algorithm.

	a_1	a_2	b_1	b_2
True	-1.914	0.949	0.226	0.1118
Mean	-1.697	0.732	0.208	0.1787
MSE	0.0475	0.0380	2.99E-05	0.0002
RMS	0.2179	0.1950	0.005468	0.0143

Table 5.3. RMSP index performance of parameter estimation.

k	$Q-ARX$	RLS
1	0.0074	2.8671
2	0.0050	2.0919
16	0.0015	0.2848
17	0.0014	0.2243
31	0.0011	0.0220
32	0.0011	0.0195

convergence speed by using Quasi-ARX model is also fast. See Fig. 5.9, the RMSP error in log scale is -2.134 or 7.4 percent can be achieved at first sampling identified by Quasi-ARX model. The same performance is reached by RLS algorithm in about forty three sampling. We can conclude that the tracking parameters estimation are very slow by using RLS algorithm.

References

[1] K.J. Aström. Maximum likelihood and prediction error methods. *Automatica*, 16:551–574, 1980.

[2] A. Keyhani, S.I. Moon, T. Leksan, and S. Sebo. System identification approach for modeling and parameter estimation of power system elements. *Electric Power Systems Research*, 32:25–33, 1995.

[3] M. Meiler, O. Schmid, M. Schudy, and E.P. Hofer. Dynamic fuel cell stack model for real-time simulation based on system identification. *Journal of Power Sources.*, 176:523–528, 2008.

[4] J.R. Rodriguez Vasquez, R. Rivas Perez, and J. Sotomayor Moriano andJ.R. Peran Gonzalez. System identification of steam pressure in a fire-tube boiler. *Computers and Chemical Engineering*, 32:2839–2848, 2008.

[5] L Ljung. *System identification, theory for the user*. Prentice-Hall, 1999.

[6] R. Johansson. *System modeling and identification*. Prentice-Hall, 1993.

[7] H.K. Sahoo, P.K. Dash, and N.P. Rath. Narx model based nonlinear dynamic system identification using low complexity neural networks and robust h_∞ filter. *Applied Soft Computing*, 13:3324–3334, 2013.

[8] T.Kara and I. Eker. Nonlinear modeling and identification of a dc motor for bidirectional operation with real time experiments. *Energy Conversion and Management*, 45:1087–116, 2004.

[9] Necla Togun, Sedat Baysec, and Tolgay Kara. Nonlinear modeling and identification of a spark ignition engine torque. *Mechanical Systems and Signal Processing*, 26:294–304, 2012.

[10] I. Eker. Open-loop and closed loop experimental on-line identification of a three mass electromechanical system. *Mechatronics.*, 14:549–565, 2004.

[11] K. S. Narendra and K. Parthasarathy. Identification and control of dynamical systems using neural networks. *IEEE Trans. on Neural Networks*, 1(1):4–27, 1990.

[12] J. Hu, K. Kumamaru, K. Inoue, and K. Hirasawa. A hybrid quasi-ARMAX modeling scheme for identification of nonlinear systems. *Trans. of the Society of Instrument and Control Engineers*, 34(8):977–985, 1998.

[13] Jinglu Hu, Kousuke Kumamaru, and Kotaro Hirasawa. A Quasi-ARMAX approach to modelling of non-linear systems. *Int. J. Control*, 74(18): 1754–1766, 2001.

[14] J. Hu and K. Hirasawa. A method for applying multilayer perceptrons to control of nonlinear systems. In *Proc. 9th International Conference on Neural Informassion Processing (Singapure)*, 2002.

[15] Y. Cheng, L. Wang, and J. Hu. Quasi-ARX wavelet network for SVR based nonlinear system identification. *Nonlinear Theory and its Applications (NOLTA), IEICE*, 2(2):165–179, 2011.

[16] Mohammad Abu Jami'in, I. Sutrisno, and J. Hu. Lyapunov learning algorithm for quasi-ARX neural network to identification of nonlinear dynamical system. In *Proc. IEEE International Conference on Systems, Man, and Cybernetics (Seoul)*, pages 3141–3146, 2012.

[17] Selami Beyhan and Musa Alci. Fuzzy functions based arx model and new fuzzy basis function models for nonlinear system identification. *Applied Soft Computing*, 10:439–444, 2010.

[18] Yajun Zhang, Tianyou Chai, Hong Wang, Jun Fu, Liyan Zhang, and Yonggang Wang. An adaptive generalized predictive control method for nonlinear systems based on ANFIS and multiple models. *IEEE Trans. on Fuzzy Systems*, 18(6):1070–1081, 2010.

[19] T. Chai, Y. Zhang, H. Wang, C. Y. Su, and J. Sun. Data-based virtual unmodeled dynamics driven multivariable nonlinear adaptive switching control. *IEEE Transactions on Neural Networks*, 22(12):2154–2172, 2011.

[20] 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*, 3(4):246–252, 2010.

[21] M.A. Jami'in, I. Sutrisno, J. Hu, Norman Bin Mariun, and Mohd Hamiruce Marhaban. Quasi-ARX neural network based adaptive predictive control for nonlinear systems. *IEEJ Trans. on Electrical and Electronic Engineering*, 11(1), 2016.

[22] C.C.Huang and C.H.Loh. Nonlinear identification of dynamic systems using neural networks. *Computer-Aided Civil and Infrastructure Engineering*, 16:28–41, 2001.

[23] Alireza Rahrooh and Scott Shepard. Identification of nonlinear systems using narmax model. *Nonlinear Analysis*, 71:1198–1202, 2009.

[24] Z. Wei, L.H. Yam, and L. Cheng. NARMAX model representation and its application to damage detection for multi-layer composites. *Composite Structures*, 68:109–117, 2005.

[25] Yang Gao and Meng Joo Er. Narmax time series model prediction: feedforward and recurrent fuzzy neural network approaches. *Fuzzy Sets and Systems*, 150:331–350, 2005.

[26] F. Abdollahi, H. A. Talebi, and R. V. Patel. Stable identification of nonlinear systems using neural networks: Theory and experiments. *IEEE/ASME Trans. on Mechatronics*, 11(4):488–495, 2006.

[27] Ben Niu, Yunlong Zhu, Xiaoxian He, and Hai Shen. A multi-swarm optimizer based fuzzy modeling approach for dynamic systems processing. *Neurocomputing*, 71:1436–1448, 2008.

[28] Y. Cheng, L. Wang, and J. Hu. Identification of Quasi-ARX neurofuzzy model with an SVR and GA approach. *IEICE Trans. Fundamentals*, E.95-A(5):876–883, 2012.

[29] M.A. Jami'in, I. Sutrisno, and J. Hu. Maximum power tracking control for a wind energy conversion system based on a Quasi-ARX neural network model. *IEEJ Trans. on Electrical and Electronic Engineering*, 10(4), 2015.

[30] Kwanghee Nam. Stabilization of feedback linearizable systems using a radial basis function network. *IEEE Trans. on Automatic Control*, 44(5): 1026–1031, 1999.

[31] Chih-Min Lin and Chun-Fei Hsu. Neural-network hybrid control for antilock braking systems. *IEEE Trans. on Neural Networks*, 14(2):351–359, 2003.

[32] Rong-Jong Wai. Hybrid fuzzy neural-network control for nonlinear motor-toggle servomechanism. *IEEE Trans. on Control Systems Tech.*, 10(4):519–532, 2002.

[33] Elkhatib Kamal, Abdelouahab Aitouche, Reza Ghorbani, and Mireille Bayart. Robust nonlinear control of wind energy conversion systems. *Int J Electr. Power and Energy Syst.*, 44:202–209, 2013.

[34] Mohammad Abu Jami'in, I. Sutrisno, and J. Hu. Nonlinear adaptive control of wind energy conversion systems based on quasi-ARX neural networks model. In *Proc. International Multi Conference of Engineers and Computer Scientis (Hongkong)*, pages 313–318, 2014.

[35] Mohammad Abu Jami'in, I. Sutrisno, and J. Hu. An adaptive predictive control based on a quasi-arx neural network model. In *Proc. 13th International Conference on Control, Automation, Robotics and Vision, Marina Bay Sands, Singapore, 10-12th December 2014 (ICARCV 2014)*, pages 253–258, 2014.

[36] Mohammad Abu Jami'in, I. Sutrisno, and J. Hu. Deep searching for parameter estimation of the linear time invariant (LTI) system by using quasi-ARX neural network. In *Proc. IEEE International Joint Conference on Neural Network (Dallas)*, pages 2759–2762, 2013.

[37] X. Yu, M. O. Efed, and O. Kaynak. A general backpropagation algorithm for feedforward neural networks learning. *IEEE Trans. Neural Netw.*, 13(1):251–254, 2002.

[38] W. Yu, A. S. Poznyak, and X. Li. Multilayer dynamic neural networks for non-linear system on-line identification. *Int. J. Contr.*, 18:1858–1864, 2001.

[39] L. Behera, S Kumar, and A. Patnaik. On adaptive learning rate that guarantees convergence in feedforward networks. *IEEE Transaction on Neural Networks*, 17(5):1116–1125, 2006.

[40] Z. Man, H. R. Wu, S. Liu, and X. Yu. A new adaptive backpropagation algorithm based on lyapunov stability theory for neural networks. *IEEE Transaction on Neural Networks*, 17(6):1580–1591, 2006.

[41] Simon Haykin. *Neural Networks, A Comprehensive Foundation*. Prentice-Hall, 1999.

[42] C. M. Lin and C. F. Hsu. Neural-network-based adaptive control for induction servomotor drive system. *IEEE Trans. Ind. Electron.*, 49(1): 238–252, 2002.

[43] Chih-Lyang Hwang and Chia-Ying Kuo. A stable adaptive fuzzy sliding-mode control for affine nonlinear systems with application to four-bar linkage systems. *IEEE Trans. on Fuzzy Systems*, 9(2):868–879, 2001.

[44] F. J. Lin, R. J. Wai, and H. P. Chen. A hybrid computed torque controller using fuzzy neural network for motor-quick-return servo mechanism. *IEEE/ASME Trans. on Mechatronics*, 6(1):75–89, 2001.

[45] Lingji Chen and Kumpati S Narendra. Nonlinear adaptive control using neural networks and multiple models. *Automatica*, 37:1245–1255, 2001.

[46] K. R. Sales and S. A. Billings. Self-tuning control of nonlinear ARMAX model. *Int. J. Control*, 51:753–769, 1990.

[47] Chi-Huang Lu. Design and application of stable predictive controller using recurrent wavelet neural networks. *IEEE Trans. on Industrial Electronics*, 56(9):3733–3742, 2009.

[48] S. Rahman and A. de Castro. Environmental impacts of electricity generation: A global perspective. *IEEE Trans. Energy Convers.*, 10(2):307–314, 1995.

[49] B. Bose. Global warming: Energy, environmental pollution and the impact of power electronics. *IEEE Ind. Electron. Mag.*, 4(1):6–17, 2010.

[50] Athanasios Mesemanolis, Christos Mademlis, and Iordanis Kioskeridis. High-efficiency control for a wind energy conversion system with induction generator. *IEEE Trans. on Energy Conv.*, 27(4):958–967, 2012.

[51] Elkhatib Kamal, Abdelouahab Aitouche, Reza Ghorbani, and Mireille Bayart. Robust fuzzy fault-tolerant control of wind energy conversion systems subject to sensor faults. *IEEE Trans. on Sust. Energy*, 3(2): 231–241, 2012.

[52] Y. She, X. She, and M. E. Baran. Universal tracking control of wind conversion system for purpose of maximum power acquisition under hierarchical control structure. *IEEE Trans. on Energy Conv.*, 26(3):766–775, 2011.

[53] E.B. Muhando, T. Senjyu, A. Yona, H. Kinjo, and T. Funabashi. Disturbance rejection by dual pitch control and self-tuning regulator for wind turbine generator parametric uncertainty compensation. *IET Control Theory Appl.*, 1:1431–1440, 2007.

[54] Roy Billinton, Rajesh Karki, YiGao, Dange Huang, Po Hu, and Wijarn Wangdee. Adequacy assessment considerations in wind integrated power systems. *IEEE Trans. Power Syst.*, 27(4):2297–2305, 2012.

[55] M. Soliman, O.P. Malik, and D.T. Westwick. Multiple model multiple-input multiple-output predictive control for variable speed variable pitch wind energy conversion systems. *IET Renew. Power Gener.*, 5(2):124–136, 2011.

[56] Yue Fu and Tianyou Chai. Nonlinear multivariable adaptive control using multiple models and neural networks. *Automatica*, 43:1101–1110, 2007.

[57] A.T. Connie, F. Ferdousi, M. Sharmin, and M.R. Khan. Identification of AR parameters at a very low SNR using estimated spectral distribution in DCT domain. *IEE Proc.-Vis. Image Signal Process.*, 153(2):95–100, 2006.

[58] Amir H. Khanshan, Hamidreza Amindavar, and Hamidreza Bakhshi. High-resolution ARMA estimation of mixed spectra. *IEEE Trans. on Signal Processing*, 58(1):97–107, 2010.

[59] Yu Hen Hu and Paul H Milenkovic. A fast least-square deconvolution algorithm for vocal tract cross section estimation. *IEEE Trans. on Acoustics. Speech. and Signal Proceesing*, 38(6):921–924, 1990.

[60] Sangit Chatterjee and Jonathan F. Bard. A comparison of box-jenkins time series models with autoregressive processes. *IEEE Trans. on Systems, Man, and Cybernetics*, 15(2):252–259, 1985.

[61] S.A. Fattah, W.-P. Zhu, and M.O. Ahmad. Identification of autoregressive moving average systems based on noise compensation in the correlation domain. *IET Signal Process.*, 5(3):292–305, 2011.

[62] William Bobillet, Roberto Diversi, Eric Grivel, Roberto Guidorzi, Mohamed Najim, and Umberto Soverini. Speech enhancement combining optimal smoothing and errors-in-variables identification of noisy AR processes. *IEEE Trans on Signal Processing*, 55(12):5564–5578, 2007.

[63] Jitendra K. Tugnait. Estimation of linear parametric models using inverse filter criteria and higher order statistics. *IEEE Trans. on Signal Processing*, 41(11):3196–3199, 1993.

[64] Rajiv M. Reddy, Issa M. S. Panahi, and Richard Briggs. Hybrid FxRLS-FxNLMS adaptive algorithm for active noise control in fMRI application. *IEEE Trans. on Control Systems Technology*, 19(2):474–480, 2011.

[65] Chee-Ming Ting, Sh-Hussain Salleh, Z. M. Zainuddin, and Arifah Bahar. Spectral estimation of nonstationary EEG using particle filtering with application to event-related desynchronization (ERD). *IEEE Trans. on Biomedical Engineering*, 58(2):321–331, 2011.

[66] Atiyeh Ghoreyshi and Henrietta L. Galiana. Simultaneous identification of oculomotor subsystems using a hybrid system approach: Introducing hybrid extended least squares. *IEEE Trans. on Biomedical Engineering*, 57(5):1089–1098, 2010.

[67] Xiangjun Zhang and Xiaolin Wu. Image interpolation by adaptive 2-D autoregressive modeling and soft-decision estimation. *IEEE Trans. on Image Processing*, 17(6):887–896, 2008.

[68] Thomas E Hall and Georgios B Giannakis. Image modeling using inverse filtering criteria with application to textures. *IEEE Trans. Image Processing*, 5(6):938–949, 1996.

[69] Young-Ho Yum and Song B. Park. Optimum recursive filtering of noisy two-dimensional data with sequential parameter identification. *IEEE Trans. on Patern Analysis and Machine Intelligence*, 5(3):337–344, 1983.

[70] Kiyong Kim, Pranesh Rao, and Jeffrey A. Burnworth. Self-tuning of the PID controller for a digital excitation control system. *IEEE Trans. on Industry Applications*, 46(4):1518–1524, 2010.

[71] Louis A. Dessaint, Bernard J. Hérbert, Hoang Le-huy, and Gianni Cavuoti. A DSP-Based adaptive controller for a smooth positioning system. *IEEE Trans. on Industrial Electronics*, 37(5):372–377, 1990.

[72] Han-Fu Chen. New approach to recursive identification for ARMAX systems. *IEEE Trans. on Automatic Control*, 55(4):868–879, 2010.

[73] Lennart Ljung. On positive real transfer functions and the convergence of some recursive schemes. *IEEE Trans. on Auitomatic Control*, AC-22(4): 539–551, 1977.

[74] Karim Nassiri-Toussi and Wei Ren. On the convergence of least squares estimates in white noise. *IEEE Trans. on Automatic Control*, 39(2):364–368, 1994.

[75] Maher Algreer, Matthew Armstrong, and Damian Giaouris. Active online system identification of switch mode DC-DC power converter based on recursive DCD-IIR adaptive filter. *IEEE Trans. on Power Electronics*, 27(11):4425–4435, 2012.

[76] X. Liu, J. Wang, and W.X. Zheng. Convergence analysis of refined instrumental variable method for continuous-time system identification. *IET Control Theory Appl.*, 5(7):868–877, 2011.

[77] M.H.H. Davis and R.B. Vinter. *Stochastic modelling and control*. Chapman and Hall, London, 1985.

[78] C. B. Feng and W.-X. Zheng. Robust identification of stochastic linear systems with correlated noise. *IEE Procedings-D*, 1991, volume = 138(5), pages = 484–492,.

[79] V. Strejc. Least squares parameter estimation. *Automatica*, 16:535–550, 1980.

[80] Feng Ding, Tongwen Chen, and Li Qiu. Bias compensation based recursive least-squares identification algorithm for MISO systems. *IEEE Trans. on Circuits and Systems-II: Express Briefs*, 53(5):349–353, 2006.