SICE Annual Conference 2015
July 28-30, 2015, Hangzhou, China

Self-Organizing Quasi-Linear ARX RBFN Modeling for Identification and Control of Nonlinear Systems

Imam Sutrisno[1,2†], Mohammad Abu Jami'in[1,2], Jinglu Hu[2] and Mohammad Hamiruce Marhaban[3]

[1]Politeknik Perkapalan Negeri Surabaya, Indonesia
(Tel: +81-80-5283-5075; E-mail: imams3jpg@moegi.waseda.jp)
[2]Graduate School of Information, Production and Systems, Waseda University, Kitakyushu Japan
(Tel: +81-93-692-5271; E-mail: jinglu@waseda.jp)
[3]CAPER, Faculty of Engineering, Universiti Putra Malaysia
(Tel: +60-12-225-8577; E-mail: mhm@upm.edu.my)

Abstract: The quasi-linear ARX radial basis function network (QARX-RBFN) model has shown good approximation ability and usefulness in nonlinear system identification and control. It owns an ARX-like structure, easy design, good generalization and strong tolerance to input noise. However, the QARX-RBFN model still needs to improve the prediction accuracy by optimizing its structure. In this paper, a novel self-organizing QARX-RBFN (SOQARX-RBFN) model is proposed to solve this problem. The proposed SOQARX-RBFN model consists of simultaneously network construction and parameter optimization. It offers two important advantages. Firstly, the hidden neurons in the SOQARX-RBFN model can be added or removed, based on the neuron activity and mutual information (MI), to achieve the appropriate network complexity and maintain overall computational efficiency for identification. Secondly, the model performance can be significantly improved through the structure optimization. Additionally, the convergence of the SOQARX-RBFN model is analyzed, and the proposed approach is applied to identify and control the nonlinear dynamical systems. Mathematical system simulations are carried out to demonstrate the effectiveness of the proposed method.

Keywords: artificial neural network, identification system, self-organization, radial basis function network

1. INTRODUCTION

Neural networks (NNs), neuro fuzzy networks (NFNs) and radial basis function networks (RBFNs) have shown highly sophisticated capability for nonlinear function approximation. However, from a user's point of view, these models are basically viewed as vehicles for adjusting the fit to the data and rarely reflect physical considerations in the system [1][2]. RBFNs are used in nonlinear system modeling and control because of their easy design, good generalization, strong tolerance to input noise, and online learning ability [3]. Recently, the quasi-linear ARX-RBFN (QARX-RBFN) model has been proposed and proved to have both universal approximation ability and easy to use linear properties in nonlinear system identification and control [4]. It owns an ARX-like linear structure, in which the coefficients are expressed by a RBFN, rather than constants. The ARX-like linear structure represents physical interpretation of applications explicitly. For example, in nonlinear adaptive control, we anticipate the prediction model is linear with current input variable for controller design, and the ARX-like structure is definitely helpful for this application [5][6][7][8].

However, the QARX-RBFN model still needs to improve the prediction accuracy by optimizing its structure, which is somewhat subjective if too simple structures of QARX-RBFN model are selected. The accuracy of the approximation provided by the QARX-RBFN model may be inadequate to achieve the desired control performance. Thus, it is important to optimize the structure of QARX-RBFN model to improve performance.

† Imam Sutrisno is the presenter of this paper.

In this paper, a novel self-organizing QARX-RBFN (SOQARX-RBFN) model is proposed to improve the prediction accuracy for nonlinear identification. The proposed SOQARX-RBFN model consists of simultaneously network construction and parameter optimization. It offers two important advantages. First, the hidden neurons in the SOQARX-RBFN model can be added or removed, based on the neuron activity and mutual information (MI), to achieve the appropriate network complexity and maintain overall computational efficiency for identification. Second, the model performance can be significantly improved through the structure optimization.

This paper begins with the self-organizing quasi-linear ARX-RBFN model in Sect. 2. In Sect. 3, a discussion and analysis about convergence of SOQARX-RBFN model is given. Two benchmark problems are simulated in Sect. 4, then discussion and conclusions are finally summarized.

2. SELF-ORGANIZING QUASI-LINEAR ARX-RBFN MODEL

2.1. System Description

Consider a single-input-single-output(SISO) nonlinear time-invariant dynamical system with input-output relation as:

$$y(t+d) = g(\varphi(t)) \tag{1}$$

$$\varphi(t)=[y(t+d-1),\cdots y(t+d-n),\, u(t),\cdots,u(t-m+1)]^T \tag{2}$$

where $y(t)$ is the output at the time $t(t = 1, 2, ...)$, $u(t)$ is the input, d is the known integer time delay, $\varphi(t)$ is the regression vector and n, m are the system orders. $g(\cdot)$ is

a nonlinear function and at a small region around $\varphi(t)$ = 0, it is C^∞ continuous, then $g(0)$ = 0.

2.2. ARX-like Predictor Expression

Applying Taylor expansion and considering the system dynamics, then the expression of the system can be described by [9]:

$$y(t+d) = \alpha(q^{-1}, \phi(t))y(t) + \beta(q^{-1}, \phi(t))u(t) \quad (3)$$

$$\alpha(q^{-1}, \phi(t)) = \alpha_{0,t} + \alpha_{1,t}q^{-1} + \cdots + \alpha_{n-1,t}q^{-n+1} \quad (4)$$

$$\beta(q^{-1}, \phi(t)) = \beta_{0,t} + \beta_{1,t}q^{-1} + \cdots + \beta_{m+d-2,t}q^{-m-d+2} \quad (5)$$

where q^{-1} is a backward shift operator, e.g. $q^{-1}u(t) = u(t-1)$ and $\phi(t) = [y(t), \cdots y(t-n+1), u(t), \cdots, u(t-m-d+2)]^T$.

This is a prediction model. It can realize the prediction of $y(t+d)$ by using the data up to t.

2.3. d-Difference Predictor Expression

Sometimes it is better to have a differential expression for controller design. For this purpose, let's consider the coefficients $\alpha_{i,t}(i = 0, \cdots, n-1)$ and $\beta_{j,t}(j = 0, \cdots, m+d-2)$ as a summation of two parts: the constant part α_i^l and β_j^l and the nonlinear function part on $\phi(t)$ which are denoted by $\alpha_{i,t} - \alpha_i^l$ and $\beta_{j,t} - \beta_j^l$ [9]. Then the expression of system in the predictor form Eq. (3) can be described by:

$$y(t+d) = \phi^T(t)\theta + \phi^T(t)\Theta_\phi^n \quad (6)$$

$$\theta = [\alpha_0^l \cdots \alpha_{n-1}^l \beta_0^l \cdots \beta_{m+d-2}^l]^T \quad (7)$$

$$\Theta_\phi^n = [(\alpha_{0,t} - \alpha_0^l) \cdots (\alpha_{n-1,t} - \alpha_{n-1}^l)(\beta_{0,t} - \beta_0^l) \cdots (\beta_{m+d-2,t} - \beta_{m+d-2}^l)]^T \quad (8)$$

where $\alpha_{i,l}, \beta_{j,l}$ are the constant parts of $\alpha_{i,t}, \beta_{j,t}$, respectively.

Applying a d-difference operator, defined by $\Delta = 1 - q^d$, to Eq. (6). Then the following expression of system in d-different form can be obtained:

$$\Delta y(t+d) = \psi^T(t)\theta + \varsigma(\Psi(t)) \quad (9)$$

where $\psi(t) = \Delta\phi(t) \cdot \varsigma(\Psi(t)) = \Psi^T(t)\tilde{\theta}_\Psi^n = \Delta\phi^T(t)\Theta_\phi^n$ and $\Psi(t) = [y(t) \cdots y(t-d-n+1)u(t) \cdots u(t-m-2d+2)]^T$.

2.4. d-Difference Predictor Expression Linear in $u(t)$

If we consider the controller design, it is better to have the prediction linear of $u(t)$ [9]. However, the Eq. (9) is a general one which is nonlinear in the variable $u(t)$, because the $\tilde{\theta}_\Psi^n$ is based on $\Psi(t)$ whose elements contain $u(t)$. To solve this problem, an extra variable $x(t)$ is introduced and an unknown nonlinear function $\rho(\xi(t))$ is used to replace the variable $u(t)$ in $\tilde{\theta}_\Psi^n$. Obviously, in a control system, the reference signal $y^*(t+d)$ can be used as the extra variable $x(t+d)$. The function $\rho(\xi(t))$ exists. Define

$$\begin{aligned}\xi(t) &= [y(t) \cdots y(t-n_1)x(t+d) \cdots \\ &\quad x(t-n_3+d)u(t-1) \cdots u(t-n_2)]^T \end{aligned} \quad (10)$$

including the extra variable $x(t+d)$ as an element. A typical choice for n_1, n_2 and n_3 in $\xi(t)$ is $n_1 = n+d-1, n_2 = m+2d-2$ and $n_3 = 0$. Then the expression of Eq. (9) can be described by:

$$\Delta y(t+d) = \psi^T(t)\theta + \Psi^T(t)\theta_\xi^n \quad (11)$$

where $\theta_\xi^n = \tilde{\theta}_\Psi^n$.

2.5. Quasi-ARX RBFN Model

By applying the parameterizations to different expressions we can have different models, and different models can be used for different applications. In order to have a model the coefficients should be parameterized, then by parameterizing α, β we can have a prediction model, by parameterizing $\alpha_{i,l}, \beta_{j,l}$ we can have a d-difference prediction model, then without losing the generality we discuss parameterizations of θ_ξ^n.

The elements of θ_ξ^n are unknown nonlinear function of $\Phi(t)$, which can be parameterized by NN or RBFN. In this paper the RBFN used has a local property,

$$\theta_\xi^n = \sum_{j=1}^{M} \mathbf{w_j}\mathbf{R_j}(\xi(\mathbf{t}), \mathbf{\Omega_j}) \quad (12)$$

where M is the number of RBFNs, $\mathbf{w_j} = [\omega_{\mathbf{1j}}, \omega_{\mathbf{2j}}, \cdots, \omega_{\mathbf{Nj}}]^\mathbf{T}$ the coefficient vector, and $\mathbf{R_j}(\xi(\mathbf{t}), \mathbf{\Omega_j})$ the RBFNs defined by:

$$\mathbf{R_j}(\xi(\mathbf{t}), \mathbf{\Omega_j}) = \mathbf{e}^{-\lambda_\mathbf{j}\|\xi(\mathbf{t}) - \mathbf{Z_j}\|^\mathbf{2}} \qquad \mathbf{j} = \mathbf{1}, \mathbf{2}, \cdots, \mathbf{M} \quad (13)$$

where $\Omega_j = \{Z_j, \lambda_j\}$ is the parameters set of the RBFN; Z_j is the center vector of RBFN and λ_j are the scaling parameters; $\|\cdot\|^2$ denotes the vector two-norm. Then we can express the quasi-ARX RBFN prediction model for Eq. (11) in a form of:

$$\Delta y(t+d) = \psi^T(t)\theta + \sum_{j=1}^{M} \Psi^T\mathbf{w_j}\mathbf{R_j}(\xi(\mathbf{t}), \mathbf{\Omega_j}) \quad (14)$$

2.6. Parameter Estimation

In order to determine the centers and widths of the RBFN, affinity propagation (AP) clustering method is employed. The center Z_j is the arithmetic mean value of all training data in each cluster. The width λ_j is ϱ times the largest distances between all training data in each cluster. The parameters θ and Θ are estimated by using on-line identification algorithms, respectively [9].

The linear parameter θ of the linear part of the model is updated as [10]:

$$\hat{\theta}(k) = \hat{\theta}(k-d) + \frac{a(t)\psi(k-d)e_1(k)}{1 + \psi(k-d)^T\psi(k-d)} \quad (15)$$

where $\hat{\theta}(k)$ is the estimate of θ at time instant k, which also denotes the parameter of a linear model used to approximate the system in d-difference form, and

$$a(t) = \begin{cases} 1 & \texttt{if } |e_1(t)| > 2D \\ 0 & \texttt{otherwise} \end{cases} \quad (16)$$

where $e_1(t)$ denotes the error of the linear model, defined by

$$e_1(t) = \Delta y(t) - \psi(t-d)^T\hat{\theta}(t-d) \quad (17)$$

2.7. Self-Organizing of QARX-RBFN

By using the parameter estimation in Section 3.2, the model structure is properly predetermined, and then a reasonable model can be obtained. The model performance can be further improved by optimizing the model structure. Then, by adding procedure to optimize the structure, we propose self-organization of QARX-RBFN. It offers two important advantages. First, the model performance can be significantly improved through the parameter optimization. Second, the hidden neurons in the SOQARX-RBFN model can be pruning or growing, based on the neuron activity and mutual information (MI), to achieve the appropriate network complexity and maintain overall computational efficiency for identification. The proposed SOQARX-RBFN model for identification approach is summarized in seven steps:

Step (1) For the unknown dynamical system Eq. (1), create an initial RBFN. The number of neurons in the input and output layers is the same as that of the input and output variables in the dynamical system. The number of neurons in the hidden layer is randomly generated. All the parameters are initialized, and the centers, widths, and connection weights of the RBFN are all uniformly distributed with a small range.

Step (2) For the input sample $u(t)$, the centers and widths are adjusted by the gradient method [11]. The output weights are trained by the formula:

$$\mathbf{w_j^T} = \eta_\mathbf{i}\mathbf{Q}(\mathbf{y(t)}, \mathbf{u(t)})\mathbf{e^T}(\mathbf{x(t)}) \tag{18}$$

where $\mathbf{Q}(\mathbf{y(t)}, \mathbf{u(t)})$ is the output value of the hidden neuron.

Step (3) Compute the active firing rate (AF), of the hidden neurons using Eq. (19). New neurons are inserted according to the activity threshold Af_o. If $Af_i > Af_o$, go to step (4), otherwise go to step (5).

$$Af_k = \frac{\rho Q_k(u)}{\sum_{k=1}^{K} Q_k(u)},\ (k = 1, 2, \cdots, K) \tag{19}$$

where Af_k is the AF value of the kth hidden neuron, K is the number of hidden neurons, Ω_k is the parameters set of the RBFN, $\rho \gg 1$ is a positive constant.

Step (4) Split the ith hidden neuron and insert new hidden neurons. The initial parameters of the newly inserted neurons are obtained from Eq. (21) and Eq. (22).

$$\mu_{k,j} = \gamma_j \mu_k + \delta_j u \tag{20}$$

$$\sigma_{k,j} = \gamma_j \sigma_k,\ j = 1, 2, \cdots, N_{new} \tag{21}$$

where $0.95 < \gamma_j < 1.05$ and $0 < \delta_j < 0.1$ (For better performance, γ_j should be close to 1, and δ_j should be close to 0.05), μ_i and σ_k are the center and width of the kth hidden neuron, and N_{new} is the number of the newly inserted neurons which is decided by the rate of active firing Af_k.

$$\omega_{k,j} = \frac{r_j(\omega_k \cdot Q_k(u) - e(u))}{Q_{k,j}(u)},\ \sum_{j=1}^{N_{new}} r_j = 1 \tag{22}$$

where r_j is the allocating parameters for the new neurons, $Q_k(u)$ is the output value of the ith hidden neuron, $Q_{k,j}(u)$ is the output value of the newly inserted jth hidden neuron, ω_k is the output weight of the kth hidden neuron, and $\omega_{k,j}$ is the output weight of the jth new hidden neuron.

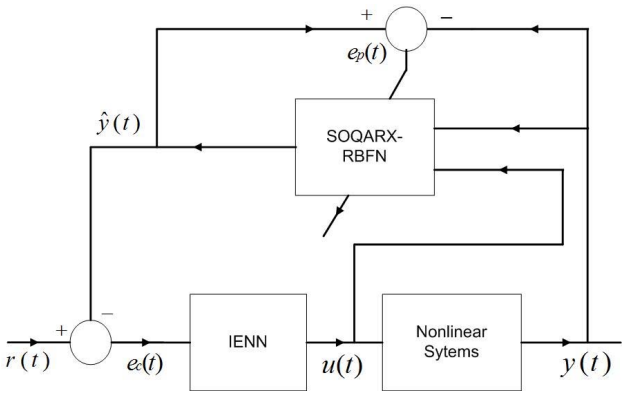

Fig. 1 IENN SOQARX-RBFN controller schema.

Step (5) Based on Eq. (23) if $m(\Omega_k; y_q)$ is less than the threshold $m_0 (0 < m_0 < 0.05)$, go to step (6). Otherwise, go to step (7).

$$m(\Omega_k; y_q) = \frac{M(\Omega_k; y_q)}{\mathtt{min}(H(\Omega_k), H(y_q))} \tag{23}$$

$$M(\Omega_k; y_q) = H(\Omega_k) - H(\Omega_k|y_q) = H(y_q) - H(y_q|\Omega_k) \tag{24}$$

where $H(\Omega_k) = \ln(2\pi\varepsilon)^K |COV(\Omega_k)|/2, COV(\Omega_k)$ is a standard covariance of Ω_k, ε is a mathematical constant here (2.718) and $H(y_q) = \ln(2\pi\varepsilon)|COV(y_q)|/2$.

Step (6) Delete the connection between the hidden neuron Ω_k and the output neuron y_q, and update the remaining RBFN parameters. Find the neuron k' in the hidden layer which has the minimal Euclidean distance between neuron k and neuron k'. The parameters of the hidden neuron k' are adjusted as follows:

$$\mu_{k',after} = \mu_{k',before}, \sigma_{k',after} = \sigma_{k',before} \tag{25}$$

$$\omega_{k',after} = \omega_{k',before} + \omega_k(\frac{Q_k(u)}{Q_{k'}(u)}) \tag{26}$$

where $\omega_{k',before}$ and $\omega_{k',after}$ are the connecting weight of the hidden neurons k' before and after structure adjustment, $\mu_{k',before}$ and $\mu_{k',after}$ are the centers of the hidden neuron k' before and after deleting neuron k and , $\omega_{k',before}$ and $\omega_{k',after}$ are the widths of the hidden neurons k' before and after deleting neuron k.

Step (7) $t = t + 1$, go to step (2). Stop when $t = T$.

3. CONTROLLER DESIGN

The improved Elman neural network (IENN) SOQARX-RBFN feedback system used in this paper is shown in Fig. 1. In Fig. 1, r, y, u and $\hat{y}$ are the reference variable, control variable (measured output), manipulated variable and predicted outputs, respectively; e_c is the error that indicates the distinction between the reference and the prediction model; e_p is the error that indicates the distinction between the process and the prediction model [14].

The purpose of IENN SOQARX-RBFN is to design a control law and an updating law for the primary controller

parameters, such that the system output y is to follow an input reference signal r and the closed-loop dynamic performance of system follows the predictive model.

The structure of the proposed IENN SOQARX-RBFN includes the input layer (i layer), the hidden layer (j layer), the context layer (r layer) and the output layer (o layer) with two inputs and one output. The basic function and the signal propagation of each layer are introduced in the following: Layer 1 (input layer): the node input and the node output are represented as:

$$X_i(k) = f_i(\mathtt{net}_i) = \mathtt{net}_i = e_i(k) \tag{27}$$

where $e_i(k)$ and $X_i(k)$ are the input and the output of the input layer, respectively and k represents the k^{th} iteration.

Layer 2 (hidden layer): the node input and the node output are represented as:

$$X_j(k) = S(\mathtt{net}_j) \tag{28}$$

$$\mathtt{net}_j = \sum_i w_{ij} X_i(k) + \sum_r w_{rj} X_r^c(k) \tag{29}$$

where $X_j(k)$ and $\mathtt{net}_j$ are the output and the input of the hidden layer, w_{ij} and w_{rj} are the connective weights of input neurons to hidden neurons and context neurons to hidden neurons, respectively, X_r^c the output of the context layer, and $S(X)$ is sigmoid function, that is, $S(X) = 1/(1+e^{-X})$.

Layer 3 (context layer): the node input and the node output are represented as:

$$X_r^c(k) = \gamma X_r^c(k-1) + X_j(k-1) \tag{30}$$

where $0 \leq \gamma < 1$ is the self-connecting feedback gain.

Layer 4 (output layer): the node input and the node output are represented as:

$$Y_o(k) = f(\mathtt{net}_o(k)) = \mathtt{net}_o(k) \tag{31}$$

$$\mathtt{net}_o(k) = \sum_j w_{jo} X_j(k) + w_o Y^c(k) \tag{32}$$

$$Y^c(k) = \zeta Y^c(k-1) + Y_o(k-1) \tag{33}$$

where $Y_o(k)$ the output of the IENN SOQARX-RBFN and also the control effort of the proposed controller, $Y^c(k)$ the output of the output feedback neuron, $0 \leq \zeta < 1$ is the self-connecting feedback gain, w_{jo} and w_o are the connective weights of hidden neurons to output neurons and output feedback neuron to output neuron, respectively.

4. DISCUSSION AND ANALYSIS ABOUT CONVERGENCE

The SOQARX-RBFN model can insert or prune the hidden nodes depending on the activities of the nodes and the connecting mutual information. This execution behavior indicates that the SOQARX-RBFN model does not guide the RBFN design process in a predefined and fixed way. Due to the activity threshold that is associated with the required error, the SOQARX-RBFN model can adjust the structure to achieve the desired learning performance. The design scheme used in the SOQARX-RBFN model is a new and efficient method of the RBFN structure design. The proposed self-organizing algorithm can be used to determine a suitable structure for improving the RBFN performance.

For the proposed SOQARX-RBFN model, the convergence of the SOQARX-RBFN model with respect to the growing, pruning and parameter learning steps is an important issue and requires careful investigation, as it is crucial for the successful applications. First, the convergence in the case without structural changes is determined. The description of convergence for parameter learning steps of the SOQARX-RBFN model can be found in [12]. Then, the convergence in the structural change phase is determined by taking into account both the growing and pruning processes.

5. SIMULATION RESULTS

To show better prediction accuracy of the proposed method, a numerical system are tested for identification of the self organizing QARX-RBFN prediction model and implementation to the nonlinear control.

5.1. Numerical System Identification

5.1.1. System Under Study

In order to study the behavior of the proposed control method, a numerical simulation is described in this section. The system is a nonlinear one governed by:

$$y(t) = f[y(t-1), y(t-2), y(t-3), u(t-1), u(t-2)] \tag{34}$$

where:

$$f[x_1, x_2, x_3, x_4, x_5] = \frac{x_1 x_2 x_3 x_5 (x_3 - 1) + x_4}{1 + x_2^2 + x_3^2} \tag{35}$$

To test the obtained model, a set of 800 input-output data is sampled as test data and the input data are described as:

$$u(t) = \begin{cases} \sin(2\pi t/250) & \text{if } t < 500 \\ 0.8\sin(2\pi t/250) & \\ +0.2\sin(2\pi t/250) & \text{otherwise.} \end{cases} \tag{36}$$

5.1.2. Results and Analysis

When identifying the system, the SOQARX RBFN prediction model Eq. (14) is used, in which the number of RBFN functions $M = 6$, the model orders $m = 3, n = 2$, the delay $d = 1$. And the bound of the nonlinear difference term of the system is set to $D = 0.05$.

Estimation of nonlinear parameter Ω_j: The nonlinear parameter vectors $\Omega_j = Z_j, \lambda_j, j = 1, \ldots, M$ are first determined off line. To do so, the system is excited by a random sequence with the amplitude between -1 and 1 as in and 1000 input-output data sets are recorded. Then an AP clustering algorithm is applied to the data set for partitioning the input space of $\xi(t) = [y(t) \ldots y(t-n) y^*(t+1) u(t-1) \ldots u(t-m)]^T$. After clustering, six clusters are generated automatically in the input space, so that

Table 1 Identification result of QARX-RBFN model for two methods.

Method	M	RMSE
QARX-RBFN	6	0.074
Self Organizing QARX-RBFN	5	0.035

$M = 6$. The parameter vector Z_j corresponds to the center of each cluster, while λ_j is calculated by multiplying a constant $\varrho = 0.2$ to the largest distance of the data in each cluster.

The final results and comparisons are shown in Table 1, in which M denotes the number of RBFN functions and the accuracy denotes model simulation root mean square error (RMSE) on the test data [13]. It is found that the value of M is fixed for the model obtained by using QARX-RBFN without self organizing and its become different for the model obtained by using QARX-RBFN with self organizing. Also it is found that the model obtained by using QARX-RBFN with self organizing gives better prediction accuracy than the model obtained by using QARX-RBFN without self organizing. The comparison simulation on the test data for identification result of QARX-RBFN model without and with self organizing gives a RMSE of 0.074 without self organizing and 0.035 with self organizing. It can be found that the identification result of QARX-RBFN model with self organizing gives better prediction accuracy than that of QARX-RBFN model without self organizing.

5.2. Nonlinear Control Implementation

5.2.1. The Desired Output

The desired output in this example is a piecewise function:

$$y^*(t) = \begin{cases} 0.6y^*(t-1) \\ +r(t-1), & t \in [1,100] \cup [151,200] \\ 0.7\mathtt{sign}(0.4493) \\ y^*(t-1) \\ +0.57r(t-1), & t \in [101,150] \end{cases} \quad (37)$$

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

5.2.2. Results and Analysis

Three main control result approaches are implemented for comparison.

- Method 1: Adaptive fuzzy switching controller based on QARX-RBFN model [4]
- Method 2: An improved Elman neural network (IENN) controller based on QARXNN [14]
- Method 3: IENN controller based on self organizing QARX-RBFN model

Figure 2 shows the control results, in which the comparisons between method 3 and method 1 are shown. In Fig. 2 the dotted line (black) is the desired output $y0(t)$, the dashed line (magenta) is the control output $y1(t)$ of method 1 and the solid line (blue) is the control output $y3(t)$ of method 3. We can easily see that the proposed method 3 has approached a good result since $t = 10$,

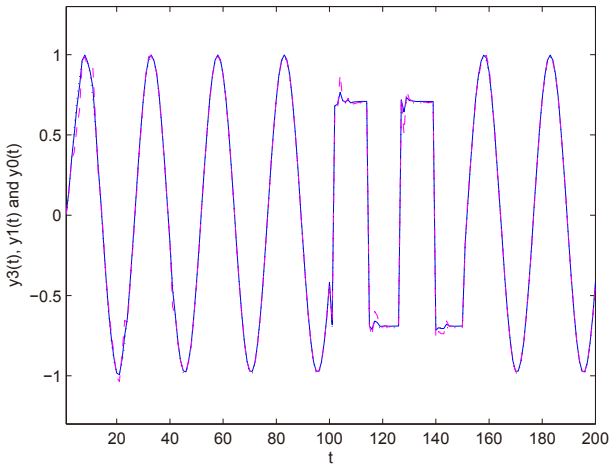

Fig. 2 Control results. IENN SOQARX-RBFN control output (solid), fuzzy switching QARX-RBFN control output (dashed) and reference (dotted)

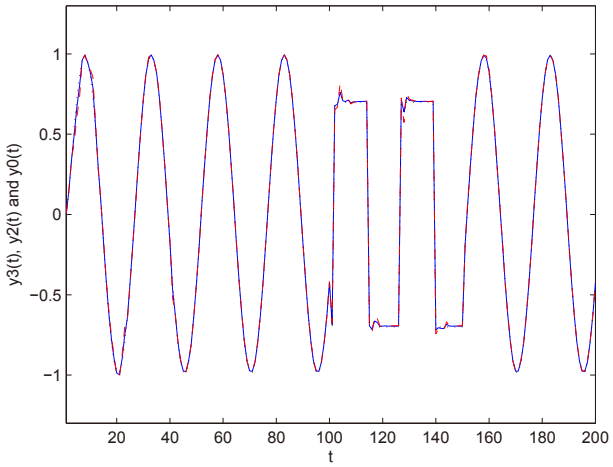

Fig. 3 Control results. IENN SOQARX-RBFN control output (solid), IENN QARXNN control output (dashed) and reference (dotted)

which is better than the method 1. It is also better than the method 1 when $t \in [(10,100) \cup (110,200)]$ and the robustness of the proposed method 3 is much better than the method 1 since $t = 100$ as illustrated. Therefore, the proposed method 3 have a better control result than the method 1. Figure 3 shows the control results, in which the comparisons between method 3 and method 2 are shown. In Fig. 3 the dotted line (black) is the desired output $y0(t)$, the dashed line (red) is the control output $y2(t)$ of method 2 and the solid line (blue) is the control output $y3(t)$ of method 3. We can easily see that the proposed method 3 has approached a good result since $t = 10$, which is better than the method 2. It is also better than the method 2 when $t \in [(10,100) \cup (110,200)]$ and the robustness of the proposed method 3 is much better than the method 2 since $t = 100$ as illustrated. Therefore, the proposed method 3 have a better control result than the method 2. Table 2 shows the results of three

Table 2 Comparison result of the errors.

Method	Mean of RMSE	Mean of var	Accuracy
Method 1	0.0141	0.057	96.24
Method 2	0.0117	0.035	97.38
Method 3	0.0075	0.011	98.46

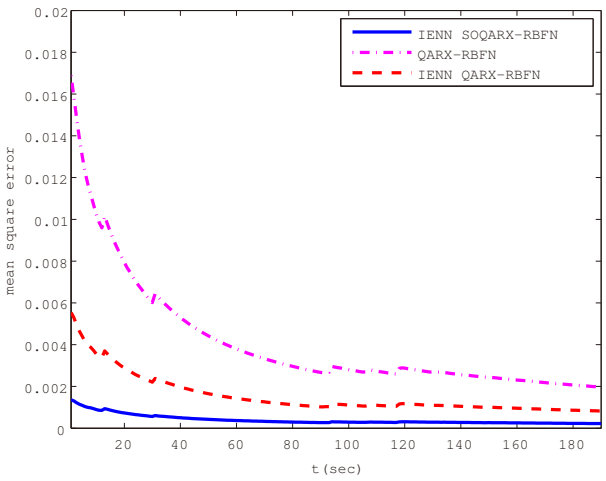

Fig. 4 Convergence characteristics of the errors.

methods. We can see that method 2 gives a smaller control error than method 1, and the method 3 improves the control performance further. For additional comparison, the convergence speed of the different methods are given in Fig. 4. The proposed method 3 gives better accuracy than the other methods.

6. CONCLUSION

In this paper, a novel SOQARX-RBFN model used to improve the prediction accuracy for nonlinear identification. The proposed SOQARX-RBFN model consists of simultaneous network construction and parameter optimization. It offers two important advantages. First, the hidden neurons in the SOQARX-RBFN model can be added or removed, based on the neuron activity and MI, to achieve the appropriate network complexity and maintain overall computational efficiency for identification. Second, the model performance can be significantly improved through the parameter optimization. The proposed SOQARX-RBFN model simplifies neural network training, and thereby significantly reduces computational complexity.

REFERENCES

[1] L. Ljung, "System Identification: Theory for The User, 2nd ed.", *Prentice-Hall PTR*, Upper Saddle River, N.J. 1999.

[2] M.A. 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 Proceedings of The 2013 International Joint Conference on Neural Networks (IJCNN2013)*, pp. 2758-2762, 2013.

[3] H. Yu, T. T. Xie, S. Paszczynski and B. M. Wilamowski, "Advantages of Radial Basis Function Networks for Dynamic System Design", *IEEE Trans. Ind. Electron*, Vol. 58, No. 12, pp. 5438-5440, 2012.

[4] L. Wang, Y. Cheng and J. Hu, "Stabilizing Switching Control for Nonlinear System Based on Quasi-ARX Model", *IEEJ Trans. On Electrical and Electronic Engineering*, Vol. 7, No. 4, pp. 390-396, 2012.

[5] J. Hu, K. Hirasawa and K. Kumamara, "A Method for Applying Neural Networks to Control of Nonlinear Systems", *Neural Information Processing: Research and Development*, pp. 351-369, Springer Berlin, Germany, 2004.

[6] I. Sutrisno, M. A. Jami'in and J. Hu, "Modified Fuzzy Adaptive Controller Applied to Nonlinear Systems Modeled under Quasi-ARX Neural Network", *Artificial Life Robotics*, Vol. 19, No. 1, pp. 22-26, 2013.

[7] I. Sutrisno, M.A. Jami'in and J. Hu, "Neural Predictive Controller of Nonlinear Systems Based on Quasi-ARX Neural Network", *in Proceedings of the 2012 International Conference on Automation and Computing (ICAC 18th 2012) (Loughborough UK)*, pp. 83-88, September 2012.

[8] M.A. Jami'in, I. Sutrisno and J. Hu, "Lyapunov Learning Algorithm for Quasi-ARX Neural Network to Identification of Nonlinear Dynamical System", *in Proceedings of The 2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC2012) (Seoul)*, pp. 3141-3146, 2012.

[9] L. Wang, "Study on adaptive control of nonlinear dynamical systems based on quasi-ARX models", *DSpace at Waseda University*, 2013.

[10] L. Chen, and K. S. Narendra, "Nonlinear Adaptive Control Using Neural Networks and Multiple Models", *Automatica*, Vol. 37, No. 8, pp. 1245-1255, 2001.

[11] J. X. Peng, K. Li and G. W. Irwin, "A Novel Continuous Forward Algorithm for RBF Neural Modelling", *IEEE Transactions on Automatic Control*, Vol. 52, No. 1, pp. 117-122, 2007.

[12] H. G. Han and J. F. Qiao, "An Adaptive Computation Algorithm for RBF Neural Network", *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 23, No. 2, pp. 342-347, 2012.

[13] I. Sutrisno, C. Che and J. Hu, "An Improved Adaptive Switching Control Based on Quasi-ARX Neural Network for Nonlinear Systems", *Artificial Life and Robotics*, Springer, Japan, Vol. 19, No. 4, pp. 347-353, October, 2014.

[14] I. Sutrisno, M. A. Jami'in and J. Hu, "An Improved Elman Neural Network Controller Based on Quasi-ARX Neural Network for Nonlinear Systems", *IEEJ Trans. On Electrical and Electronic Engineering*, Vol. 9, No. 5, pp. 494-501, 2014.