Hierarchical Algorithms of Quasi-Linear ARX Neural Networks for Identification of Nonlinear Systems

Mohammad Abu Jami'in, *Member, IAENG,* Yuyun, *Member, IAENG,* Eko Julianto, *Member, IAENG*

***Abstract*—A quasi-linear ARX neural network model (QARXNN) is a nonlinear model built using neural networks (NN). It has a linear-ARX structure where NN is an embedded system to give the parameters for the regression vector. There are two contributions in this paper, 1) Hierarchical Algorithms is proposed for the training of QARXNN model, 2) an adaptive learning is implemented to update learning rate in NN training to ensure global minima. First, the system is estimated by using LSE algorithm. Second, nonlinear sub-model performed using NN is trained to refine error of LSE algorithm. The linear parameters obtained from LSE algorithm is set as bias vector for the output nodes of NN. Global minima point is reached by adjusting the learning rate based on Lyapunov stability theory to ensure convergence of error. The proposed algorithm is used for the identification and prediction of nonlinear dynamic systems. Finally, the experiments and numerical simulations reveal that the proposed method gives satisfactory results.**

***Index Terms*—System Identification, quasi linear-ARX neural network, hierarchical algorithm, convergence of error, nonlinear system**

I. Introduction

SYSTEM identification is an extensive problem because it is difficult to propose a model capable of describing efficiently in every possible nonlinear system. Neural networks (NN) and neuro-fuzzy networks are a type of black-box models with nonparametric or multi-parametric models [1]. The Black-box models for the identification of nonlinear systems do not require parametric expressions. The most useful property of NN is its ability to approximate arbitrary linear or nonlinear mapping through training. 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, 2) it is difficult to use their parameters for controller design, and 3) the stability analysis is not easy, and the parameter tuning is generally a time-consuming process due to its nonlinear and multi-parametric nature [2], [3].

Some researchers adopted hybrid approach by mixing linear and nonlinear models to improve accuracy, simplicity and reducing computational complexity [4], [1], [5], [6]. A hybrid fuzzy time series model is constructed based on ARIMA and Interval Type-2 Fuzzy Inference System was proposed to improve forecasting accuracy [4]. In this paper, we propose a quasi-linear ARX neural network model (QARXNN). It has an easy-to-use structure shown by its ARX-like structure. Nonlinear coefficients are implemented by NN to parameterize the input vector. It is injected to the QARXNN model as an embedded system [1]. This is the interesting aspect of the structure that the coefficients have some meaning to describe the dynamic behavior of the identified plants. The parameters of system modeling can be used to estimate the stability of the modeled system and can be set as controller parameters [8], [9]. The coefficients consist of linear and nonlinear parts that allow for system analysis using the approach of linear theory.

In the previous research, the embedded system of the QARXNN model was modified by using neuro-fuzzy, wavelet, radial-basis function, and multilayer perceptron (MLP) networks in order to increase the accuracy of model for system identification [10]. However, this work does not discuss the stable and global convergence of system identification. In this paper, a hierarchical algorithm was proposed for the training of QARXNN model based on Lyapunov stability theory. The learning rate is updated every leaning step in order to guarantee the convergence of error. Thus, the error will be asymptotically goes to zero by the number of training time.

Lyapunov stability is a well-known theory for control application to guarantee closed-loop control stability. A lyapunov-like analysis was proposed to derive stable learning law in order to ensure the stability property of the identification error [11]. Using the Lyapunov stability theorem for the training algorithm allows for increasing the speed of the convergence in the learning process, guaranteeing global convergence of the model, and avoiding local minima and achieving global minima of cost function [12], [13]. The Lyapunov function has been used to track convergence of error; the output of tracking error can then asymptotically converge to zero [13] and can be used to analyze the stability of the identification process with stable learning law [14], [15], [16].

In this paper, the training of QARXNN model is performed hierarchically between linear and nonlinear sub-models based on Lyapunov stability theorem. First, the system is estimated using a linear sub-model under LSE algorithm. From this step, we have linear part parameters. Second, the residual errors of LSE algorithm (linear sub-model) are refined by nonlinear sub-model performed using NN. The parameters of linear part is set as bias vector for the output nodes of NN, where the residual errors of linear sub-model is set as the target output to train NN sub-model. Thus, the errors of the linear sub-model is refined by the training of nonlinear

Manuscript received 05 April, 2017; revised 19 July, 2017.

Mohammad Abu Jami'in is with the Department of Automation Engineering, Politeknik Perkapalan Negeri Surabaya, Surabaya, Jawa Timur, Indonesia. e-mail: jammysby@gmail.com.

Yuyun is with the Department of Computer Engineering, STMIK Handayani, Makassar, Sulawesi Selatan, Indonesia.

Eko Julianto is with the Department of Piping Engineering, Politeknik Perkapalan Negeri Surabaya, Surabaya, Jawa Timur, Indonesia.

sub-model. From the training of NN, we have nonlinear part parameters. Thus, by summing between the linear and nonlinear parts parameters, we have nonlinear parameters. If the system is linear then the coefficients converge to the specific value of the constant, and if the system is nonlinear then the coefficients will be a function of time [17]. Especially in control applications, this approach allow us to derive the linear or nonlinear controllers based on the linear or nonlinear parameters. A simple switching mechanism is introduced between the linear and nonlinear parts parameters in order to derive linear and nonlinear controls. [8], [18], [7].

The rest of the paper is organized as follows: Section 2 contains a brief discussion of the QARXNN model. In Section 3, the proposed hierarchical learning algorithm is derived with linear and nonlinear sub-models. Convergence analysis of the proposed algorithm is also described in this section. Experimental results are presented in Section 4, and finally conclusions are recorded in Section 5.

II. Quasi-Linear ARX Neural Network Model

Consider a single-input-single-output (SISO) black-box nonlinear system whose input-output relation is described by

$$y(t) = g(\phi(t)) \quad (1)$$

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. The general nonlinear system expressed in (1) can be represented in a regression which has been shown in Ref.[1,2]. The unknown nonlinear function $g(\phi(t))$ can be performed using Taylor expansion on a small region of around $\phi(t)$ = 0:

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

As (2) shows, the coefficients are infinite. However, Taylor expansion is only used as a bridge to transform the nonlinear system. Applying the Taylor series transformation and the system dynamics, the nonlinear system can be expressed as [1]:

$$y(t) = y(0) + \phi(t)^T\Omega(\phi(t)). \quad (3)$$

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

$$\Omega(\phi(t)) = [\dot{g}(0)\phi(t) + \frac{1}{2}\phi^T(t)\ddot{g}(0)\phi(t) + \cdots]^T. \quad (4)$$
$$= [a_{1,t}\ \cdots\ a_{n_u,t}\ b_{0,t}\ \cdots\ b_{n_y-1,t}] \quad (5)$$

where, $a_{i,t} = a_i(\phi(t)), (i = 1, \cdots, n_y)$ is the coefficient of the regression vector of output and $b_{j,t} = b_j(\phi(t)), (j = 0, \cdots, n_u - 1)$ is the coefficient of the regression vector of input. The orders of time delay for the system input-output are n_u and n_y.

The Quasi-linear ARX neural network is a flexible nonlinear model that has linear properties [19]. Fig. 1 shows the structure of quasi-linear ARX neural network model. Selecting NN as an embedded system, the model can be written as follows:

$$y(t, \phi(t)) = \phi(t)^T\aleph(\phi(t), \Omega) \quad (6)$$
$$= \phi(t)^T W_2\Gamma W_1(\phi(t)) + \phi(t)^T\theta, \quad (7)$$

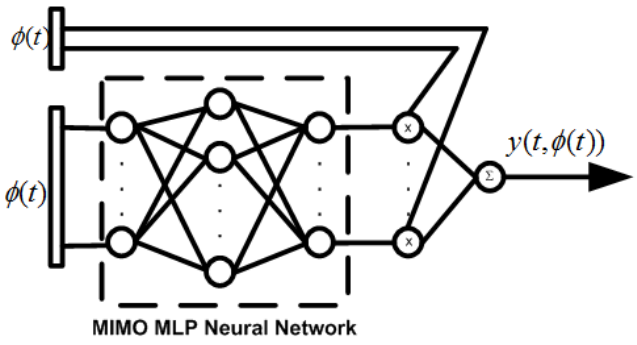

Fig. 1. A quasi-linear ARX neural network model.

where $\Omega = \{W_1, W_2, \theta\}$ sets the network parameters. The network parameter is Γ, W_1, W_2, and θ, being the diagonal nonlinear operator with 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 coefficients or state dependent parameter estimation (SDPE), denoted by $\aleph(\phi(t), \Omega)$ is implemented using NN stated as

$$\aleph(\phi(t), \Omega) = W_2\Gamma W_1(\phi(t)) + \theta \quad (8)$$
$$= \delta(\phi(t), W) + \theta. \quad (9)$$

As this demonstrates, when compared to conventional feed-forward neural networks, the quasi-linear ARX neural network has an easy-to-use structure and introduces efficient algorithms for the parameter estimation as shown by (7) and (8). The quasi-linear ARX neural networks model consists of two parts: the second term of the right side of (7) shows the linear part, while the first term is the nonlinear part. The coefficients of the linear and nonlinear parts are fused to parameterize the input vector. Error in the linear and nonlinear sub-models is shared. In one step, both linear and nonlinear parts are updated simultaneously. The estimated parameter of the linear part changes at every learning step. Thus, it is difficult to analyze the system identified linearly using the linear part.

III. Hierarchical Algorithm with Adaptive Learning for QARXNN Model

In this paper, we use a different approach to describe the QARXNN model by separating the linear and nonlinear sub-models. First, the linear sub-model is used to identify the system to obtain the linear parameter estimate. The linear estimator allows for the calculation of the linear parameter estimation and the residual error of the model. This feature allow for analyzing the identified plant as a linear system. Second, the residual error of the linear sub-model is used to train NN. NN is a compensator to refine the error of the linear sub-model. The proposed hierarchical algorithm fixed the linear parameter estimator, which does not change in any training step of NN. Interestingly, this method can be used to describe the system linearly and nonlinearly via the switching mechanism [8]. In order to perform the hierarchical

algorithm, the Equation (7) can be rewritten as:

$$\begin{aligned} y(t,\phi(t)) &= z_l + z_n && (10)\\ z_l &= \phi^T(t)\theta && (11)\\ z_n &= \phi^T(t)\delta(\phi(t),W) && (12)\end{aligned}$$

where, z_l is the linear estimator (linear sub-model) and z_n is the nonlinear estimator (nonlinear sub-model).

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

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

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

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

By using the QARXNN model, the estimated output of (7) can be rewritten in the following form:

$$\hat{y}(t) = \phi^T(t)\hat{\aleph}(\phi(t),\Omega) \quad (13)$$

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

$$y(t) = \phi^T(t)\aleph^*(\phi(t),\Omega) \quad (14)$$

Using (13) and (14), the adaptation error $e(t)$ is derived as follows:

$$\begin{aligned} e(t) &= y(t)-\hat{y}(t)\\ &= \phi^T(t)\aleph^*(\phi(t),\Omega) - \phi^T(t)\hat{\aleph}(\phi(t),\Omega)\\ &= \phi^T(t)\Xi && (15)\\ \Xi &= \aleph^*(\phi(t),\Omega) - \hat{\aleph}(\phi(t),\Omega). && (16)\end{aligned}$$

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 accuracy and convergence of error characteristics are then greatly influenced by the adaptation error Ξ.

The adaptation error is then distributed to the two sub-models by

$$\begin{aligned} e_l &= y(t) - \phi^T(t)\hat{\theta} && (17)\\ e_n &= y(t) - \phi^T(t)\hat{\delta}(\phi(t),W). && (18)\end{aligned}$$

where $\hat{\theta}$ denotes the linear part-parameter estimator and $\hat{\delta}(\phi(t),W)$ denotes nonlinear part-parameter estimator. The residual error of the linear and nonlinear sub-models are e_l and e_n, respectively. The linear part-parameter estimator is the parameter for the linear system. Therefore, it is possible to analyze the system in a linear fashion. Hereafter, e_l can be set as the target output for the nonlinear sub-model z_n. By introducing two linear and nonlinear sub-models, equation (15) can be rewritten in the following form:

$$\begin{aligned} e(t) &= y(t)-\hat{y}(t)\\ &= y(t) - \phi^T(t)\hat{\aleph}(\phi(t),\Omega)\\ &= y(t) - \phi^T(t)(\hat{\delta}(\phi(t),W)+\hat{\theta})\\ &= y(t) - \phi^T(t)\hat{\delta}(\phi(t),W) - \phi^T(t)\hat{\theta}\\ &= e_l - z_n && (19)\\ &= e_n - z_l. && (20)\end{aligned}$$

The desired goal of system identification is to make the adaptation error $e(t)$ reach and remain at zero. Two algorithms are implemented hierarchically. First, the system is estimated using the linear sub-model under the LSE algorithm to obtain linear parameter estimator θ. Second, the residual error of the linear sub-model is calculated by (17) and set as the target output for the nonlinear sub-model performed by a multilayer perceptron neural network (MLPNN). Because the linear parameter estimator is fixed and set as a bias vector for the output nodes of NN, sub-model z_n is estimated by (12) until its output converge to e_l. By (19), $e(t)$ will converge to zero if the output of nonlinear sub-model z_n converges to e_l. Thus, the nonlinear sub-model becomes a booster to refine the estimator of the linear sub-model.

To train the NN for the nonlinear sub-model, adaptive learning is proposed for training similar to a BP algorithm. The fixed learning rate of BP is replaced by the adaptive learning rate performed based on the Lyapunov stability theorem. The selected Lyapunov function is defined as follows:

$$V = \frac{1}{2}e^2(k) \quad (21)$$

where $e(k) = \{e_l^1 - z_n^1, e_l^2 - z_n^2, \cdots, e_l^N - z_n^N\}^T$, N is the number of training data, and k is the learning sequence. Performing the derivative of (21) in discrete form, we have

$$\begin{aligned} \Delta V(k) &= V(k+1) - V(k)\\ &= \frac{1}{2}e^2(k+1) - e^2(k) && (22)\end{aligned}$$

By introducing the increment of error,

$$e(k+1) = e(k) + \Delta e(k) \quad (23)$$

Substituting (23) in (22), we have

$$\begin{aligned} \Delta V(k) &= \frac{1}{2}\left(\ (e(k)+\Delta e(k))^2 - e^2(k)\ \right)\\ &= \frac{1}{2}(\Delta e(k))^2 + e(k)\Delta e(k) && (24)\end{aligned}$$

The parameters of weight W are updated using the gradient descent algorithm that may be defined as

$$\begin{aligned} W(k+1) &= W(k) + \Delta W\\ &= W(k) + \eta\left(\ -\frac{\partial V}{\partial W}\ \right)\\ &= W(k) + \eta\left(\ \frac{\partial z_n}{\partial W}\ \right)e(k). && (25)\end{aligned}$$

Thus, ΔW can be rewritten as

$$\Delta W = \eta\left(\ \frac{\partial z_n}{\partial W}\ \right)e(k) \quad (26)$$

For learning, the difference of error is due to the change during learning, which can be expressed as

$$\begin{aligned} \Delta e(k) &\approx \frac{\partial e}{\partial W}\\ &\approx -\left(\ \frac{\partial z_n}{\partial W}\ \right)^T \Delta W && (27)\end{aligned}$$

Let $\frac{\partial z_n}{\partial W} = H$. By using (26) and (27) in (24), we have

$$\begin{aligned} \Delta V(k) &= (-H\Delta W)^2 + e(k)(-H\Delta W(k))\\ &= H^2(\eta H e(k))^2 + e(k)(-H\eta H e(k))\\ &= \eta^2 H^4 e^2(k) - \eta H^2 e^2(k) && (28)\end{aligned}$$

According to the Lyapunov stability theorem, the condition of stability is dependent on a negative $\Delta V(k)$. Therefore, the trajectory of $\Delta V(k)$ is as follows:

$$\Delta V(k) = -\lambda e^2(k) \quad (29)$$

Substituting (28) and (29), we have

$$\lambda = \eta H^2 - \eta^2 H^4 \quad (30)$$

Based on the Lyapunov stability theorem, the stability is ensured if $V(k)$) is positive-definite and if $\Delta V(k)$ is negative-definite. Therefore, the learning rate η is adapted to guarantee a global convergence condition, which can be satisfied if $\lambda > 0$. Therefore, the learning rate is adapted to satisfy the condition as follows:

$$\eta \left(\frac{\partial z_n}{\partial W} \right)^2 < 1 \quad (31)$$

The Lyapunov function is selected with the positive definite function that represents the energy function. In each learning step, the parameters of the network is updated based on reducing the energy, and the energy is always dissipated. The global convergence point of cost function is ensured if the energy and dissipated energy converge to zero. This can be shown by both as the error and the gradient of error converge to zero through training. To guarantee global convergence under Lyapunov stability theory, the derivative of the Lyapunov function is ensured in that negative definite function.

The detailed steps to train a QARXNN model based on Lyapunov theorem is described as follows:

1) Set $y(t)$ as z_l then estimate θ using LSE algorithm of (11).
2) Calculate e_l using (17) and set as output guide to train sub-model z_n in (12).
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 using adaptive learning based on Lyapunov function (ALLF). 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 (25) and learning condition satisfy (31)
7) Stop if pre-specified condition is met, otherwise goto step 3, set $k = k + 1$

The parameters of the prediction model obtained from system identification is tested to predict the output of d ahead prediction as follows:

$$\hat{y}(t+d) = \phi^T(t+d)\hat{\aleph}(\phi(t+d), \Omega) \quad (32)$$

where, $\hat{\aleph}(\phi(t+d), \Omega) = [\hat{a}_{(1,t+d)} \cdots \hat{a}_{(n_y,t+d)} \quad \hat{b}_{(1,t+d)} \cdots \hat{b}_{(n_u,t+d)}]^T$ and $\phi(t+d) = [y(t+d-1)\ y(t+d-2) \cdots y(t+d-n_y)\ u(t+d-1)\ u(t+d-2) \cdots u(t+d-n_u)]^T$ are SDPE and the input vector of d step ahead prediction, respectively.

IV. Simulation Studies

In order to evaluate the performance of the proposed method, a number of simulation studies are carried out for identification and prediction of nonlinear systems. The plant models are taken from the literature in order to complete a performance comparison.

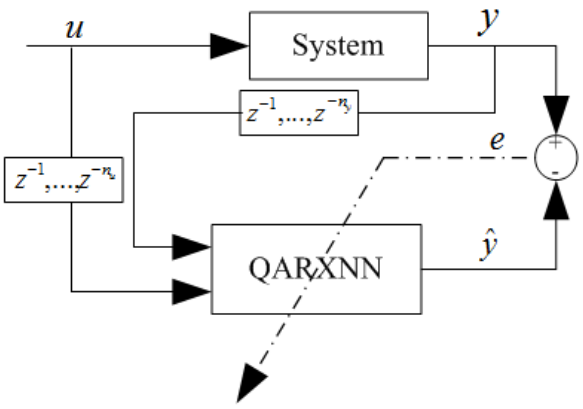

Fig. 2. Identification scheme using QARXNN model.

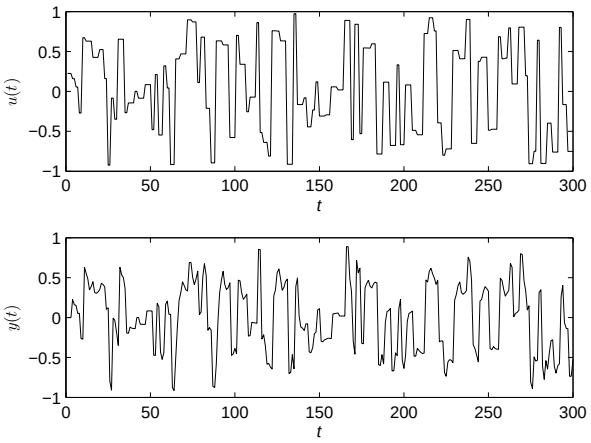

Fig. 3. System's input-output of training data.

A. System Identification Example 1

System identification is used to find the relationship between the input and output of the system. The structure of the system identification is shown in Fig. 2. The input of the system modeling is the input u and output y composed with time delay, and $\hat{y}$ is the estimated output of QARXNN model. The QARXNN model parameters are updated according to estimation error e. A nonlinear dynamic system taken from Narendra [20] is used in this example as follows:
$y(t) = f(y(t-1), y(t-2), y(t-3), u(t-1), u(t-2))$

$$f(x_1, x_2, x_3, x_4, x_5) = \frac{x_1 x_2 x_3 x_5 (x_3 - 1) + x_4}{1 + x_2^2 + x_3^2}, \quad (33)$$

A number of input-output training data is obtained by testing the system with a pseudo random binary sequence (PRBS) function with the amplitude [-1.0;1.0]. The first 300 training data are shown in Fig. 3. We set the model parameters by $\aleph_{(5,5,5,1)}$ which demonstrates that the model contains five elements of input $\phi(t) = [y(t-1)y(t-2)y(t-3)u(t-1)u(t-2)]^T$, five hidden nodes, five output nodes of the embedded system and one output node. Thus, QARXNN model has 55 parameters.

An index of normalized prediction error (NPE) and root mean square (RMS) error are introduced in order to measure the performance of system identification and prediction is

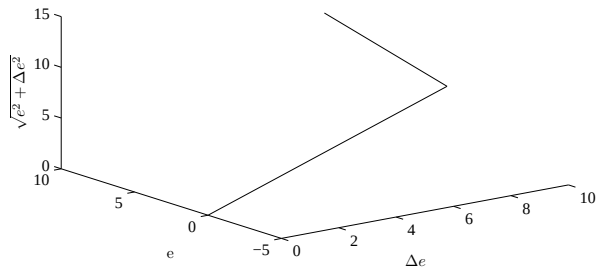

Fig. 4. Learning pattern of QARXNN model with the proposed method.

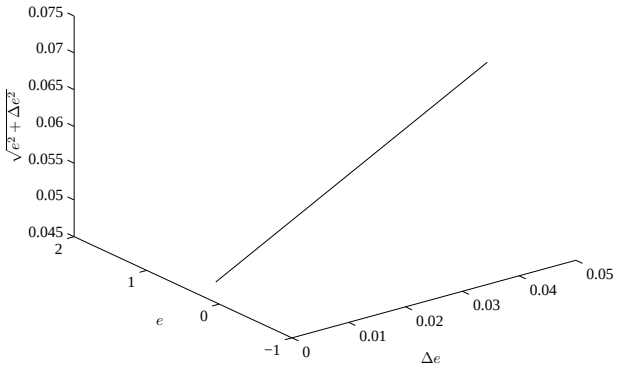

Fig. 5. Learning pattern of QARXNN model with original BP method.

stated as follows:

$$NPE = \left(\frac{\sum_{t=1}^{N}(y(t)-\hat{y}(t))^2}{\sum_{t=1}^{N}(y(t))^2}\right)^{1/2} x100\%. \quad (34)$$

$$RMS = \sqrt{\frac{\sum_{t=1}^{N}(y(t)-\hat{y}(t))^2}{N}} \quad (35)$$

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.

The learning pattern of the proposed method and the original BP method have been illustrated in Fig. 4 and Fig. 5. As we can see in Fig. 4 and Fig. 5, the error and gradient error of learning contribute in training the model in order to obtain optimal weights, whereas BP only contributes by learning error. According to the Lyapunov stability theorem, the global minimum point of cost function can be guaranteed if the learning error and the derivative of the error converge to zero.

The model obtained from the system identification is tested with the input-output of the training data to estimate next step prediction shown in Fig. 6 and Fig. 7. The training data has uniform distribution. Therefore, the NPE index prediction using the proposed method is consistent to the specific value of 21.7118 for all times. Compared with the BP method that gives an average NPE=30.6449 with standard deviation 0.0389. Thus, the proposed method is able to ensure a stable identification error.

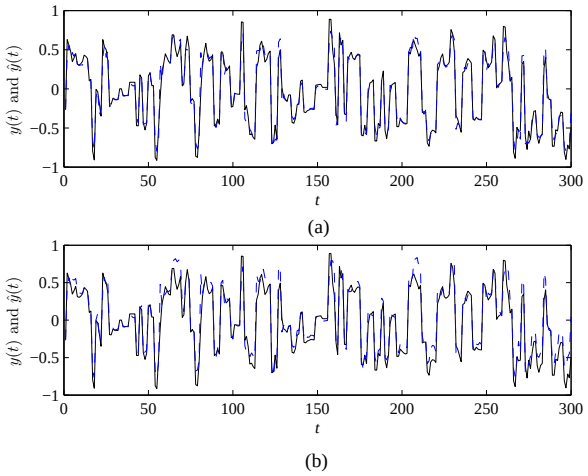

Fig. 6. Output of one-step-ahead prediction (a) the proposed method. (b) QARXNN-BP method.

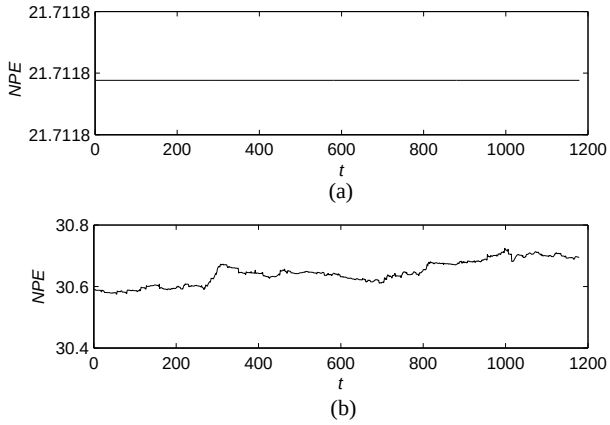

Fig. 7. NPE index of one-step-ahead prediction. (a) the proposed method. (b) QARXNN-BP method.

After training, the test signal $u(t)$ is fed to the model as follows:

$$u(t) = \begin{cases} \sin(2\pi t/250), & \text{if } t \leq 500 \\ 0.8\sin(2\pi t/250) & \\ +0.2\sin(2\pi t/25), & \text{if } t > 500 \end{cases} \quad (36)$$

Fig. 8, Fig. 9 and Fig. 10 show the results of the simulation using the tested signal. The accuracy of the proposed method is 0.0297, whereas the BP method is 0.0792. Compared to the previous work shown in Tab.I, it can be seen that the proposed method performs better than the other methods.

TABLE I
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

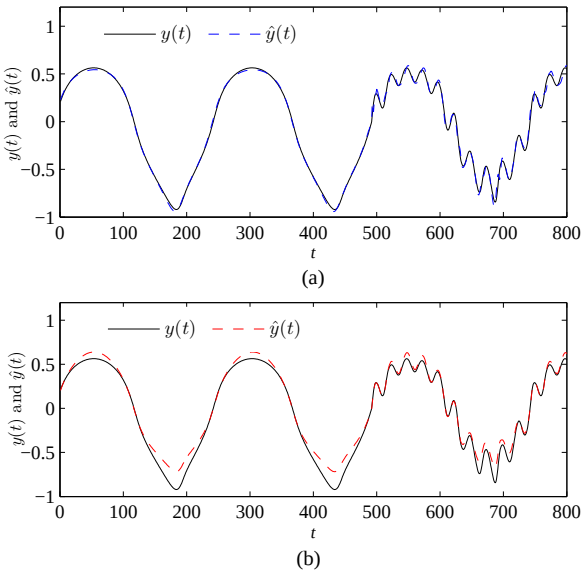

Fig. 8. Simulation results with the deterministic signal (a) the proposed method. (b) QARXNN-BP method

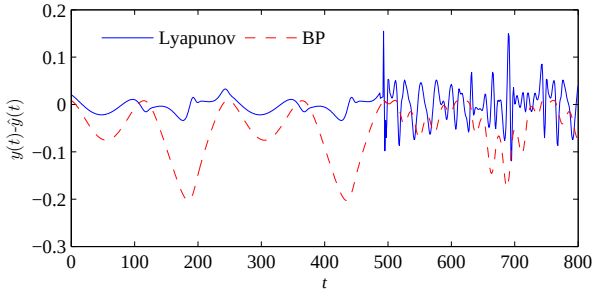

Fig. 9. The error of simulation tested with deterministic signal.

B. System Identification Example 2

This example considers the modeling of the nonlinear system to be as follows:

$$\begin{aligned} y(t) &= 0.72y(t-1) + 0.025y(t-2)u(t-2) \\ &+ 0.01u^2(t-3) + 0.2u(t-4). \end{aligned} \quad (37)$$

The same mathematical model of dynamic systems is also used in [21], [22], [23]. The current output of the system depends on two previous output values and three previous input values. However, only two values, $y(t-1)$ and $u(t-1)$,

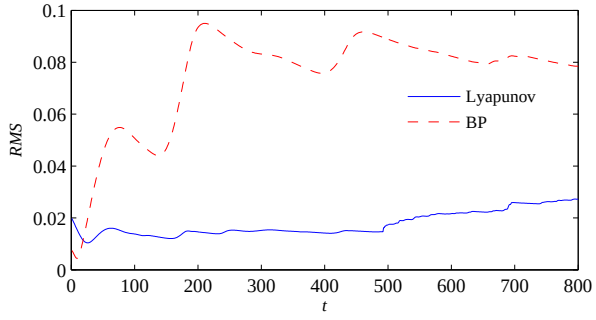

Fig. 10. RMS errors of the simulation tested with deterministic signal.

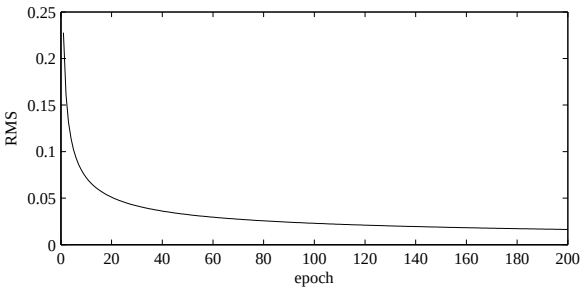

Fig. 11. RMS values obtained during training.

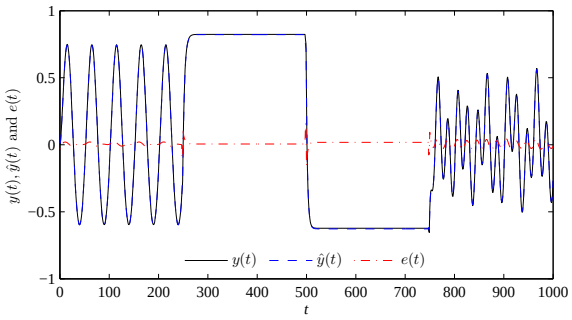

Fig. 12. The results of identification

are fed to the QARXNN model in order to compare it to other models. The length of input of training data is 900. Half of the inputs are independently and identically distributed (i.i.d.) in a uniform sequence over [-2,2], and the remaining are sinusoid generated by $1.05\sin(\pi t/45)$. QARXNN model is trained for 200 epochs, as shown in Fig. 11. After training, the following same test signal $u(t)$ is used for testing the performance of the QARXNN model:

$$u(t) = \begin{cases} \sin(\pi t/25), & \text{if } t < 250 \\ 1.0, & \text{if } 250 \le t < 500 \\ -1.0, & \text{if } 500 \le t < 750 \\ 0.3\sin(\pi t/25) & \\ +0.1\sin(\pi t/32) & \\ +0.6\sin(\pi t/10), & \text{if } 750 \le t < 1000 \end{cases} \quad (38)$$

The parameters of the model are denoted as $\aleph_{(2,4,2,1)}$, which shows the model contains two elements of input, four hidden nodes, two output nodes of the embedded system and one output node . Thus, QARXNN model has 20 parameters. Fig. 12 compares the actual output of the system with that of the QARXNN model. The RMS values of the tested model are shown in Fig. 13. A detailed performance comparison of the QARXNN model is given in Tab.II.

TABLE II
COMPARISON OF QARXNN MODEL WITH OTHER MODELS

Models	Number of parameters	RMS train	RMS test
RSONFIN [24]	49	0.03	0.06
TRFN-S [21]	33	0.0067	0.0313
FWNN [23]	27	0.01973	0.022609
FWNN [23]	43	0.018713	0.020169
AWN [22]	12	0.009368	0.022933
AWN [22]	20	0.009391	0.023259
Q-ARX-NN-ALLF	20	0.01637	0.01999

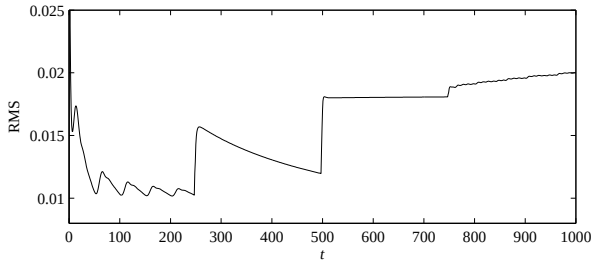

Fig. 13. RMS values of tested model

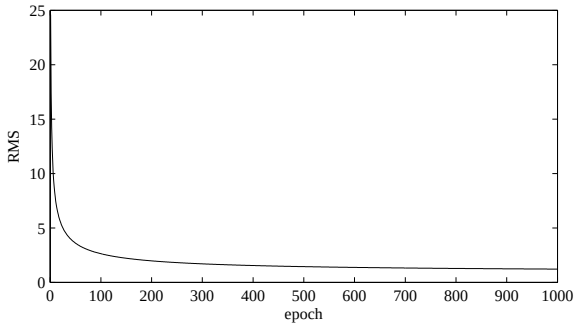

Fig. 14. RMS values obtained during training.

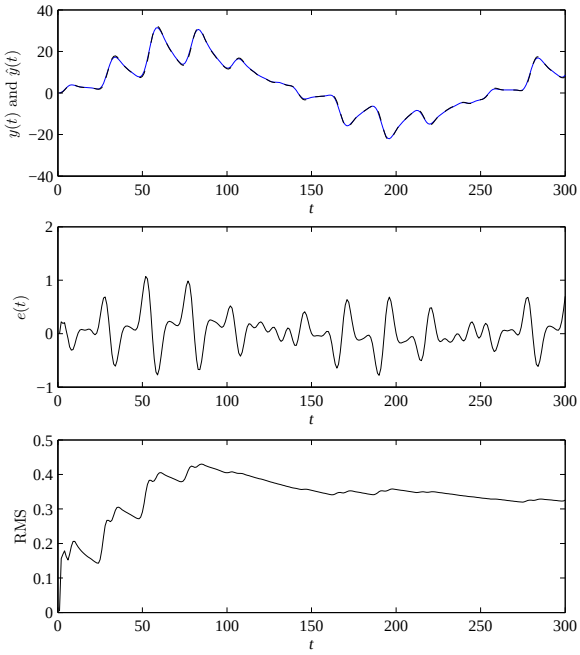

Fig. 15. Testing the model with training data.

C. Prediction of nonlinear system

This example considers a nonlinear second order dynamical model that has been used in [20], [25], [16]. The process is described by the following equation:

$$y(t) = 0.3y(t-1) + 0.6y(t-2) + f[u(t)] \tag{39}$$

where

$$f[u(t)] = u^3(t) + 0.3u^2(t) - 0.4u(t) \tag{40}$$

$$u(t) = sin(2\pi t/250) + sin(2\pi t/25). \tag{41}$$

The function $'f'$ is a polynomial with three degree of current input signal $u(t)$. The 500 input-output data is generated. Three hundred data are used in training procedure and the remaining 200 data-points are for prediction. The network parameters of QARXNN model used is $\aleph_{(3,3,3,1)}$. The model contains twenty one parameters with three elements of input, three hidden nodes, three output nodes of embedded system and one output node.

The models are trained with 1000 epochs, as shown in Fig. 14. The results of the system identification in which the model is tested with training data is shown in Fig. 15, which also shows that the RMS value is 0.3254. Fig. 16 contains the results of prediction when the adaptation of the trained model stops after 300 samplings. As we can see, the proposed model is able to predict a nonlinear system where the RMS index for prediction is 0.343.

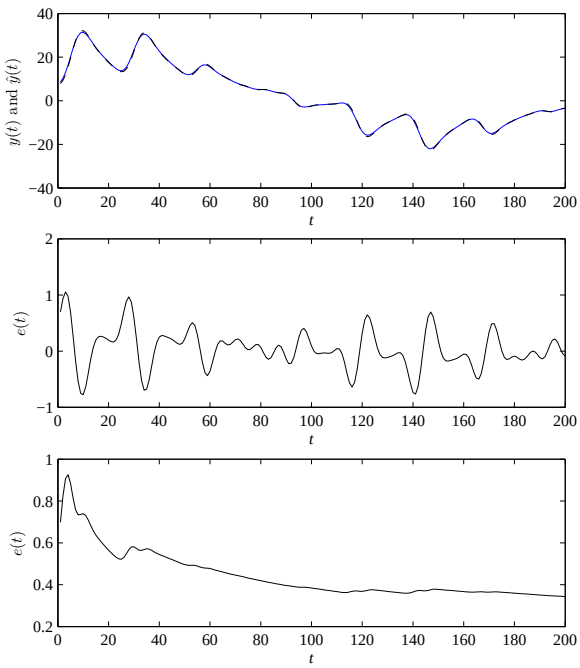

Fig. 16. Prediction output after training model stops at t = 300.

V. Discussion and Conclusion

This paper discusses two issues: 1) a different approach to the hierarchical algorithm of linear and nonlinear sub-models of the quasi-linear ARX neural network is introduced. First, the system is identified using a linear estimator by LSE algorithm. Second, the residual error of the linear sub-model is set as a target output performed using NN. 2) In order to improve the performance of system identification and prediction, we propose using the adaptive learning performed based on Lyapunov stability theorem. The fixed learning rate of BP is replaced by the adaptive learning rate based on Lyapunov stability theory. By using a Lyapunov function-based adaptive learning approach to the system embedded in the QARXNN model, the proposed method is able to improve the performance of system identification and prediction.

REFERENCES

[1] J. Hu, K. Kumamaru and K. Hirasawa, "A Quasi-ARMAX approach to modelling of nonlinear systems," *Int. J. Control*, vol. 74, no. 18, pp 1754-1766, 2001

[2] E. Kamal, A. Aitouche, R. Ghorbani and M. Bayart, "Robust nonlinear control of wind energy conversion systems," *Int. J. Electr. Power and Energy Syst.*, vol. 44, pp 202-209, 2013

[3] Y. Cheng, L. Wang and J. Hu, "Identification of Quasi-ARX neuro-fuzzy model with an SVR and GA approach," *IEICE Trans. Fundamentals*,vol. E.95-A, no. 5, pp 876-883, 2012

[4] S. Hassan, J. Jaafar, B.S. Brahim and A.J. Tahseen, "A Hybrid Fuzzy Time Series Model for Forecasting," *Engineering Letters*, vol. 20, no. 1, pp 88-93, 2012

[5] G. P. Zhang, "Time series forecasting using a hybrid ARMA and neural network model," *Neurocomputing*, vol. 50, pp 159-175, 2003

[6] Kuan-Heng Chen and Khaldoun Khashanah, "Analysis of Systemic Risk: A Vine Copula-based ARMA-GARCH Model," *Engineering Letters*, vol. 24, no. 3, pp 268-273, 2016

[7] M.A. Jami'in, I. Sutrisno and J. Hu, "Nonlinear Adaptive Control for Wind Energy Conversion Systems Based on Quasi-ARX Neural Networks Model," *Lecture Notes in Engineering and Computer Science: Proceedings of The International MultiConference of Engineers and Computer Scientists 2014*, IMECS 2014, 12-14 March, 2014, Hong Kong, pp 313-318.

[8] M.A. Jami'in, I Sutrisno, J. Hu, N.B. Mariun and M.H. Marhaban, "An adaptive predictive control based on a quasi-ARX neural network model," *Proc. 13th International Conference on Control, Automation, Robotics and Vision*, pp 253-258, Dec. 2014

[9] M.A. Jami'in, I Sutrisnoa, J. Hu, N.B. Mariun and M.H. Marhaban, "Quasi-ARX Neural Network Based Adaptive Predictive Control for Nonlinear Systems," *IEEJ Transaction on Electrical and Electronic Engineering*, vol. 11, no. 1, pp 83-90, 2016

[10] Y. Cheng, L. Wang and J. Hu, "Quasi-ARX wavelet network for SVR based nonlinear system identification," *Nonlinear Theory and its Applications (NOLTA), IEICE*, vol. 2, no. 2, pp 165-179, 2011

[11] W Yu, A.S. Poznyak, X. Li, "Multilayer dynamic neural networks for non-linear system on-line identification," *Int. J. Control*, vol. 74, no. 18, pp 1858-1864, 2001

[12] L. Bahera, S. Kumar and A. Patnaik, "On adaptive learning rate that guarantees convergence in feedforward networks," *IEEE Transaction on Neural Networks*, vol. 17, no. 5, pp 1116-1125, 2006

[13] 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*, vol. 17, no. 6, pp 1580-1591, 2006

[14] V. Becera, F Garces, S. Nasuto and W Holderbaum, "An efficient parameterization of dynamic neural networks for nonlinear system identification," *IEEE Transaction on Neural Networks*, vol. 16, no. 4, pp 983-988, 2005

[15] 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*, vol. 11, no. 4, pp 488-495, 2006

[16] M.A. Shoorehdeli, M. Teshnehlab and A.K. Sedigh, "Identification using ANFIS with intelligent hybrid stable learning algorithm approaches," *Neural Computing and Applications*, vol. 18, pp 157-174, 2009

[17] 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," *Proc. IEEE International Joint Conference on Neural Network*, pp 2758-2762, 2013

[18] M.A. Jamiin, J. Hu and E. Julianto, "A Lyapunov Based Switching Control to Track Maximum Power Point of WECS," *Proc. IEEE International Joint Conference on Neural Network*, pp 3883-3888, 2016

[19] J. Hu and K. Hirasawa, *A method for applying neural networks to control of nonlinear systems*, in book entitled Neural Information Processing: Research and Development, Springer, Berlin, Germany, pp 351-369, 2004

[20] K.S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Trans. on Neural Networks*, vol. 1, no. 1, pp 4-27, 1990

[21] C.F. Juang, "A TSK-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithms," *IEEE Trans.Fuzzy Syst.*, vol. 10, no. 2, pp 155-170, 2002

[22] Y. Oysal and S Yilmaz, "An adaptive wavelet network for function learning," *Neural Computing and Applications*, vol. 19, pp 383-392, 2010

[23] R.H. Abiyev, O. Kaynak, "Fuzzy wavelet neural networks for identification and control of dynamic plants a novel structure and a comparative study," *IEEE Trans. Ind. Electron*, vol. 55, no. 8, pp 3133-3140, 2008

[24] C.F. Juang and C.T. Lin, "A recurrent self-organizing neural fuzy inference network," *IEEE Trans. Neural Network*, vol. 10, no. 4, pp 828-845, 1999

[25] A. Banakar, M.F. Azeem, "Artificial wavelet neural network and its application in neuro-fuzzy models," *Appl. Soft Comput.*, vol. 8, no. 4, pp 1463-1485, 2008

Mohammad Abu Jami'in received the M.E. degree in Control Engineering from Institut Teknologi Sepuluh Nopember (ITS) Surabaya, Indonesia in 2008, and the Doctor of Engineering in Neurocomputing from Waseda University, Japan in 2016. He is currently a Lecturer with the Politeknik Perkapalan Negeri Surabaya, Indonesia. His research interests include artificial intelligence and its applications.

Yuyun received the B.E. degree in Computer Engineering STMIK Dipanegara Makassar and the M.E. degree in Electrical Engineering from Hasanuddin University Makassar, Sulawesi Selatan, Indonesia in 2010 and 2013. He is currently a lecture in STMIK Handayani Makassar, Indonesia. His research interests include Computer Engineering.

Eko Julianto received the B.E. degree in Marine Engineering from Institut Teknologi Sepuluh Nopember (ITS) Surabaya, Indonesia in 1989, and the M.E degree in Marine Engineering from University Of New Castle Upon Tyne, UK, 1994. He is currently a Lecturer with the Politeknik Perkapalan Negeri Surabaya, Indonesia. His research interests include Marine Engineering.