Proceedings of the 18th International Conference on Automation & Computing, Loughborough University, Leicestershire, UK, 8 September 2012

Neural Predictive Controller of Nonlinear Systems Based on Quasi-ARX Neural Network

Imam Sutrisno, Mohammad Abu Jami'in, and Jinglu Hu
Graduate School of Information Production and Systems
Waseda University
Kitakyushu, Japan
E-mail: imams3jpg@moegi.waseda.jp, mohammad@ruri.waseda.jp, jinglu@waseda.jp

Abstract—**This paper present a neural predictive controller (NPC) based on improved quasi-ARX neural network (IQARXNN) for nonlinear dynamical systems. The IQARXNN is used as a model identifier with switching algorithm and switching stability analysis. The primary controller is designed based on a modified Elman neural network (MENN) controller using back-propagation (BP) learning algorithm with modified particle swarm optimization (MPSO) to adjust the learning rates in the BP process to improve the learning capability. The adaptive learning rates of the controller are investigated via Lyapunov stability theorem, which are respectively used to guarantee the convergences of the predictive controller. Performance of the proposed MENN controller with MPSO is verified by simulation results to show the effectiveness of the proposed method both on stability and accuracy.**

Keywords-neural predictive controller (NPC); improved quasi-ARX neural network (IQARXNN); modified Elman neural network (MENN); modified particle swarm optimization (MPSO); stability and accuracy

I. Introduction

The NPC of nonlinear dynamical systems has attracted much attention and developed significantly over the past few decades. Some researchers have successfully applied neural network (NN) as the model identifier of the NPC system for controlling the real processes; for examples, Piche *et al.* [1] presented NN-based modeling and control techniques by developing nonlinear dynamic models from empirical input-output data, Zeng *et al.* [2] established a NN predictive control scheme for studying the coagulation process of wastewater treatment in a paper mill, and Wang *et al.* [3] presented adaptive NN model based predictive control for air-fuel ratio of SI engines.

The RNN is a dynamical mapping and demonstrates good control performance in the presence of unmodelled dynamics; each recurrent neuron has an internal feedback loop, and then captures the dynamic response of a system without external feedback through delays [4][5]. Some researchers have extensively investigated RNN-based predictive control with its application to nonlinear systems; for examples Parlos *et al.* [6] presented an architecture for integrating NN with industrial controllers for use in predictive control of complex process systems, Huang and Lewis [7] developed and analyzed a RNN predictive feedback control structure for a class of uncertain continuous-time nonlinear dynamic time-delay system in a canonical form.

The MENN as one kind of RNN were proposed to improve the dynamic characteristic and convergence speed [8][9]. A MENN approximation-based computed-torque controller is proposed to deal with unmodelled bounded disturbances and unstructured unmodelled dynamics of the robot arm [8]. An improved Elman NN was developed to realize failure detection in a hydraulic servo system [9].

There are two major criticisms on neural network models. The first one is that their parameters do not have useful interpretations. The second one is that they do not have a friendly interface for controller design and system analysis [10][11][12]. To solve these problems, a quasi-ARX neural network model has been proposed which embodied a macro-model part and a kernel part [10][13]. The quasi-ARX neural network can be used to identify nonlinear systems accurately.

In this paper, the quasi-ARX neural network is divided into two parts: the linear part is used to ensure the nonlinear stability, and the nonlinear part is utilized to improve the accuracy. In order to combine both the stability and universal approximation capability in our controller, a switching law is established based on system input-output variables and prediction errors.

The existing work and the previous works of authors [19][20][21][22][23] is developing controller of nonlinear systems based on quasi-ARX neural network, and current work are focus on neural predictive controller.

In this paper, a MENN controller with MPSO is developed to control nonlinear systems based on an improved quasi-ARX prediction model. In Section 2, the considered system is given. In Section 3, an improved quasi-ARX prediction model is introduced based on a NN with a switching law and analyzes the stability. Section 4 describes the MENN controller using MPSO for learning rates adjustment and investigated by Lyapunov stability theorem. Numerical simulations are carried out to show the effectiveness of the proposed model in Section 5. At last Section 6 gives some conclusion.

II. Problem Formulation

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

$$y(t)=g(\varphi(t))+v(t). \quad (1)$$

$$\varphi(t)=[y(t-1), \ldots, y(t-n), u(t-d), \ldots, u(t-m-d+1)]^T$$

where $y(t)$ as output, $u(t)$ as input at time t ($t = 1, 2, \ldots$), d the recognized integer time delay, $\varphi(t)$ the regression vector, and $v(t)$ the system noise and $g(.)$ is a nonlinear function.

The next assumptions will be used:

Assumption 1:

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

(ii) there is a rational indefinite controller which may be expressed by $u(t)=\square(\square(t))$ definition $\square(t)=[y(t)\ldots y(t-n)u(t-1)\ldots u(t-m)y^*(t+1) \ldots y^*(t+1-l)]^T$ ($y^*(t)$ is reference output);

III. Improved Quasi-ARX Neural Network

A. Quasi-ARX Neural Network

Through using Taylor expansion for (1), a similar-linear ARX model could be developed as

$$A(q^{-1}, \phi(t))y(t) = y_0+ B(q^{-1}, \phi(t))\, q^{-d}\, u(t-1) + v(t). \quad (2)$$

where $A(q^{-1},\phi(t))$ and $B(q^{-1},\phi(t))$ is defined by:

$$A(q^{-1},\phi(t)) = 1 - a_{1,t}\, q^{-1} - \ldots - a_{n,t}\, q^{-n}$$

$$B(q^{-1},\phi(t)) = b_{0,t} + \ldots + b_{m-1,t}\, q^{-m+1}$$

$$\phi(t) == [y(t) \ \ldots y(t-n+1)u(t) \ldots u(t-m-d+2)]^T$$

Introducing the following marks:

$$\psi(t) = [1\ y(t) \ \ldots y(t-n+1)\ u(t) \ldots u(t-m-d+2)]^T$$

$$\Phi(\square(t),\chi(t)) = [y_{\square,\chi}\ \alpha_{0,t} \ldots \alpha_{ny-1,t}\ \beta 0,t \ldots \beta_{nu+d-2}]^T,$$

Finally the improved quasi-ARX NN expression by :

$$y_p(t+d|t, \square(t))= \psi^T(t)\, \Phi(\square(t), \chi(t)). \quad (3)$$

B. Switching Law

The switching law define by the following function

$$J_i(t)=\sum_{l=k}^{(t)}\frac{a_i(l)(\|e_i(l)\|^2-4\Delta^2)}{2(1+a_i(l)\Psi(l-k)^T\Psi(l-k))}+c\sum_{l=t-N+1}^{t}(1-a_i(l)\|e_i(l)\|^2)$$

i=1,2. (4)

Then, the expression of switching law $\chi(t)$ described as $\chi(t)=1$ if $J_1(t) > J_2(t)$ otherwise $\chi(t)=0$. If $J_1(t) > J_2(t)$ the nonlinear part is added, else only use linear part to identify.

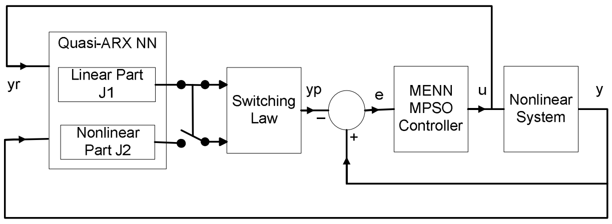

Figure 1. NPC structure by neural network.

IV. Design Of NPC

A. NPC Scheme

The purpose of NPC is to design a control law and an updating law for the primary controller parameters, such that the system output y is following input reference signal yr and the closed-loop dynamic performance of system is following predictive model. The interior characteristic of NPC guarantees the dynamic performance of the closed-loop system as long as the controller is properly designed.

Fig. 1 show the NPC scheme which will be used in the proposed nonlinear system. A switching mechanism is employed to improve the performance of the quasi-ARX neural network prediction model. The primary controller is modified Elman neural network controller with modified particle swarm optimization.

B. Modified Elman Neural Network Controler Design

Modified Elman neural network is better than common Elman neural network because the feedback of the output layer is taken into account, so better learning efficiency can be obtained. In a common Elman neural network, the hidden layer neurons are fed by the outputs of the context neurons and the input neurons.

Context neurons are memory units as they store the previous output of hidden neurons. Because a common Elman neural network only employs the hidden context nodes to diverse message, it has low learning speed and convergence precision.

Fig. 2 show the structure of the proposed MENN including 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 [14].

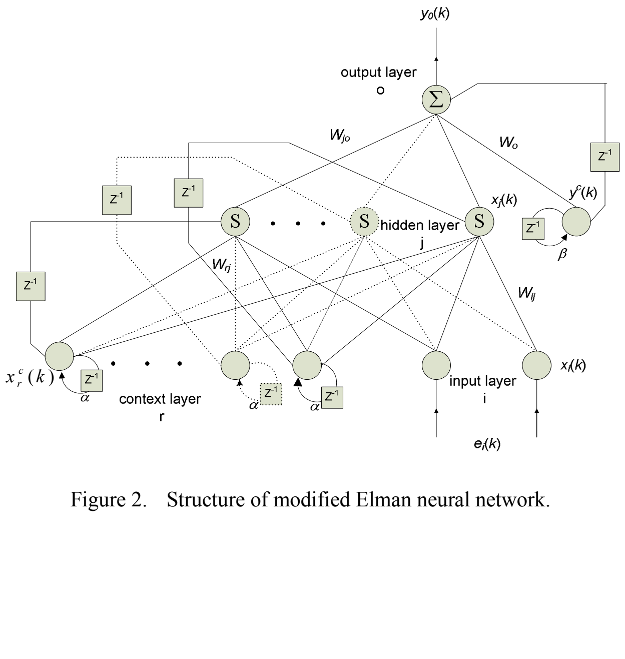

Figure 2. Structure of modified Elman neural network.

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

$$x_i(k) = f_i(\mathrm{net}_i) = \mathrm{net}_i = e_i(k). \tag{5}$$

where $e_i(k)$ and $x_i(k)$ are the input and the output of the input layer, respectively, k represents the kth iteration.

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

$$x_j(k) = S(\mathrm{net}_j). \tag{6}$$

$$net_j = \sum_i W_{ij} \times x_i(k) + \sum_r W_{rj} \times x_r^c(k) \cdot \tag{7}$$

where $x_j(k)$ and net_j are the output and the input of the hidden layer, W_{ij} and W_{rj} the connective weights of input neurons to hidden neurons and context neurons to hidden neurons, respectively, $x_r^c(k)$ 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

$$x_r^c(k) = \alpha x_r^c(k-1) + x_j(k-1) \cdot \tag{8}$$

where $0 \le \alpha < 1$ is the self-connecting feedback gain.

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

$$y_o(k) = f(\mathrm{net}_o(k)) = \mathrm{net}_o(k). \tag{9}$$

$$net_o(k) = \sum_j W_{jo} \times x_j(k) + W_o \times y^c(k) \cdot \tag{10}$$

$$y^c(k) = \beta y^c(k\text{-}1) + y_o(k\text{-}1). \tag{11}$$

where $y_o(k)$ the output of the modified Elman neural network and also the control effort of the proposed controller, $y^c(k)$ the output of the output feedback neuron, $0 \le \beta < 1$ 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.

C. Learning Algorithm

The learning algorithm of the modified Elman neural network is referred to as the BP learning rule method and chain rule. To describe using supervised gradient decent method, first the energy function E is defined as

$$E = ½\,(y - y_p)^2 = ½\,\varepsilon_o^2. \tag{12}$$

where y and y_p represents the output of the system and output of the prediction model, respectively, and ε_o denotes the tracking error.

Then the learning algorithm is defined as follows:

Layer 4: the error term to be propagated is given by

$$\delta_o = -\frac{\partial E}{\partial y_o(k)} = -\frac{\partial E}{\partial \varepsilon_o}\frac{\partial \varepsilon_o}{\partial y_o(k)} = -\frac{\partial E}{\partial \varepsilon_o}\frac{\partial \varepsilon_o}{\partial y_p}\frac{\partial y_p}{\partial y_o(k)} \cdot \tag{13}$$

$$\begin{aligned}\Delta W_o &= -\eta_1 \frac{\partial E}{\partial W_o} = -\eta_1 \frac{\partial E}{\partial y_o(k)}\frac{\partial y_o(k)}{\partial net_o(k)}\frac{\partial net_o(k)}{\partial W_o} \cdot \\ &= \eta_1 \delta_o y^c(k)\end{aligned} \tag{14}$$

$$\begin{aligned}\Delta W_{jo} &= -\eta_2 \frac{\partial E}{\partial W_{jo}} = -\eta_2 \frac{\partial E}{\partial y_o(k)}\frac{\partial y_o(k)}{\partial net_o(k)}\frac{\partial net_o(k)}{\partial W_{jo}} \cdot \\ &= \eta_2 \delta_o x_j(k)\end{aligned} \tag{15}$$

The connective weights W_o and W_{jo} are updated according to the following equations, correspondingly

$$W_o(k+1) = W_o(k) + \Delta W_o. \tag{16}$$

$$W_{jo}(k+1) = W_{jo}(k) + \Delta W_{jo}. \tag{17}$$

where the factors η_1 and η_2 are the learning rate.

Layer 3: through using chain rule, the update law of W_{rj} is

$$\begin{aligned}\Delta W_{rj} &= -\eta_3 \frac{\partial E}{\partial W_{rj}} = -\eta_3 \frac{\partial E}{\partial y_o(k)}\frac{\partial y_o(k)}{\partial net_o(k)}\frac{\partial net_o(k)}{\partial x_j(k)}\frac{\partial x_j(k)}{\partial W_{rj}} \\ &= \eta_3 \delta_o W_{jo} x_j(k)[1 - x_j(k)] x_r^c(k)\end{aligned} \tag{18}$$

$$W_{rj}(k+1) = W_{rj}(k) + \Delta W_{rj}. \tag{19}$$

Layer 2: through using chain rule, the update law of W_{ij} is

$$\begin{aligned}\Delta W_{ij} &= -\eta_4 \frac{\partial E}{\partial W_{ij}} = -\eta_4 \frac{\partial E}{\partial y_o(k)}\frac{\partial y_o(k)}{\partial net_o(k)}\frac{\partial net_o(k)}{\partial x_j(k)}\frac{\partial x_j(k)}{\partial W_{ij}} \\ &= \eta_4 \delta_o W_{jo} x_j(k)[1 - x_j(k)] x_i(k)\end{aligned} \tag{20}$$

$$W_{ij}(k+1) = W_{ij}(k) + \Delta W_{ij}. \tag{21}$$

where the factors η_3 and η_4 are the learning rate, (η_1, η_2, η_3, η_4) will be optimized by the MPSO algorithm.

Finally, a delta adaptation law as follows is adopted because of the Jacobian of the unknown dynamics system is difficult to be determined [15]

$$\delta_o \cong (y - y_p) + (\dot{y} - \dot{y}_p) = \varepsilon_o + \dot{\varepsilon}_o \cdot \tag{22}$$

where y and y_p represent the first derivatives of the output of the system and output of the prediction model.

D. Learning Rates Adjustment using MPSO

PSO is an algorithm based on a group of flying birds to simulate the behaviour of a swarm as a simplified collective system.

To further improve the learning ability, MPSO algorithm is adopted for tuning the learning rates η_1, η_2, η_3, and η_4 of the MENN [16][17][18].

1) Step 1: initialization

With $R_i^d = [R_i^1, R_i^2, R_i^3, R_i^4]$ for learning rates (η_1, η_2, η_3, η_4), the population size is set to P=15, and the dimension of particle is set to d=4. The parameters needed to be optimized with minimum and maximum ranges.

2) Step 2: Define location and velocity

Initial location $R_i^d(N)$ and velocities $v_i^d(N)$ of all particles are randomly generated in the search space. The elements in vector $R_i^d(N)$ are randomly generated by

$$R_i^d \sim U[\eta_{\min}^d, \eta_{\max}^d]\,. \tag{23}$$

where $U[\eta_{\min}^d, \eta_{\max}^d]$ designate the result of uniformly distributed random variable ranging over the known lower and upper bounded values $\eta_{\min}$ and $\eta_{\max}$ of the learning rate.

3) Step 3: Update velocity

Every particle in the swarm is updated using (24). The velocity update law is described as

$$\begin{aligned} v_i^d(N+1) &= w v_i^d(N) + c_1 \cdot r_1 \cdot (pbest_i^d - R_i^d(N)) \\ &+ c_2 \cdot r_2 \cdot (gbest_i^d - R_i^d(N)) \end{aligned}\,. \tag{24}$$

where pseudorandom sequences $r_1 \sim U(0,1)$ and $r_2 \sim U(0,1)$ are used to simulate the stochastic nature of the algorithm. For all dimensions d, let $R_i^d, pbest_i^d$ be the current position and current personal best position.

4) Step 4: Update position

The new velocity is then added to the current position of the particle to find its next position by

$$R_i^d(N+1) = R_i^d(N) + v_i^d(N+1) \quad i = 1,\ldots,P\,. \tag{25}$$

5) Step 5: Update pbests

If the current position of a particle is located within the analysis space and does not intrude the territory of other gbests, the objective function of the particle is evaluated. The fitness of each particle is calculated by

$$FIT = \frac{1}{0.1 + abs(y_p - y)}\,. \tag{26}$$

by using (25), a gradually increasing fitness function can be obtained.

6) Step 6: Update gbests

The *g*best is replaced by the best *p*best among the particles. Each particle R_i^d memorizes its own fitness value and chooses the maximum one that is the best so far its defined as $pbest_i^d$ and the maximum vector in the population $pbest_i^d = [pbest_1^d, pbest_2^d, \cdots, pbest_p^d]$ is obtained.

7) Step7: Check convergence

Steps 3-6 are repeated until the best fitness value for the *g*best is obviously improved or a set count of the generation is reached. Finally the highest fitness value $gbest_i^d$ is the optimal learning rate (η_1, η_2, η_3, η_4) of MENN. The acceleration coefficients c_1 and c_2 can be used to control the move distance of a particle in a single iteration, typically set to 2.0 for simplicity. The inertia weight w in (27) is used to control the convergence behavior of the MPSO. Small values of w result in a more rapid convergence usually on a suboptimal position, while large values may cause divergence. In general, the inertia weight w is set according to

$$w = w_{\max} - \frac{w_{\max} - w_{\min}}{iter_{\max}} \cdot iter\,. \tag{27}$$

where iter$_{\max}$ is the maximum number of iterations, and the *iter* is the current iteration count. The maximum values of the inertia weights are $w_{\max}$=0.5 and $w_{\min}$ =0.3, respectively.

E. Stability Analysis for the NPC based on Lyapunov

The following theorem states that the NPC controller using MENN with MPSO is also convergent based on Lyapunov stability theory.

Theorem Let the weights of MENN are updated along with (14), (15), (18) and (20). Then the convergence of the MENN (11) is guaranteed if the learning rate η satisfies is chosen to satisfy

$$\eta = \frac{\beta \left[\sum_{o=1}^{m_y} \varepsilon_o(k) \frac{\partial y_p(k)}{\partial W}\right]^T \left[\sum_{o=1}^{m_y} \varepsilon_o(k) \frac{\partial y_p(k)}{\partial W}\right]}{\left[\sum_{o=1}^{m_y} \frac{\partial y_p(k)}{\partial W}\right]^T \left[\sum_{o=1}^{m_y} \varepsilon_o(k) \frac{\partial y_p(k)}{\partial W}\right]}\,. \tag{28}$$

where the condition are $0 < \beta < 2$ and
$W = [W_{11}^o \cdots W_{m_y n_h}^o \quad W_{11}^I \cdots W_{n_h n_i}^I \quad W_1^H \cdots W_{n_h}^H]^T$.

Proof. Let a Lyapunov function candidate be chosen as $l(k) = \sum_{o=1}^{m_y} \varepsilon_o^2(k)$, and let $\partial l(k) = l(k+1) - l(k)$ and $\partial \varepsilon_o(k) \equiv \varepsilon_o(k+1) - \varepsilon_o(k)$. Then

$$\partial l(k) = 2\sum_{0=1}^{m_y} \varepsilon_o(k) \partial \varepsilon_o(k) + \sum_{0=1}^{m_y} (\partial \varepsilon_o(k))^2\,. \tag{29}$$

By the method in [24], $\delta\varepsilon_o(k)$ can be represented by $\delta\varepsilon_o(k)=[\delta\varepsilon_o(k)/\delta W]^T \delta W$, and (29) becomes

$$\begin{aligned} \partial l(k) = -\eta &\left(2\left[\sum_{o=1}^{m_y} \varepsilon_o(k) \frac{\delta y_p(k)}{\delta W}\right]^T \left[\sum_{o=1}^{m_y} \varepsilon_o(k) \frac{\delta y_p(k)}{\delta W}\right]\right. \\ -\eta &\left.\left(\left[\sum_{o=1}^{m_y} \frac{\delta y_p(k)}{\delta W}\right]^T \left[\sum_{o=1}^{m_y} \varepsilon_o(k) \frac{\delta y_p(k)}{\delta W}\right]\right)^2\right) \end{aligned}\,. \tag{30}$$

To ensure this selected learning rate inside the stable region, we set the learning rate η as in (28) and then have $\delta l(k) < 0$ which shows that the MENN is convergent. This completes the proof of the theorem.

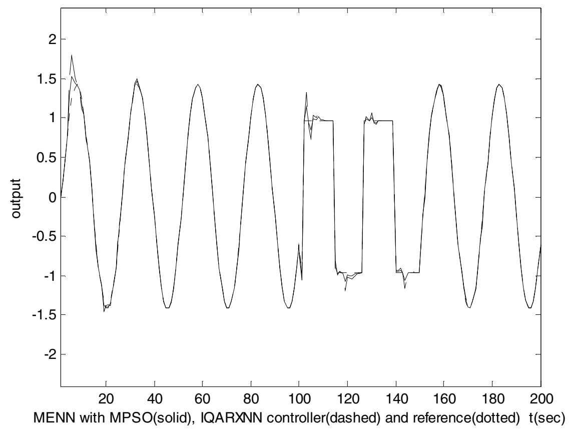

Figure 3. Simulation result of proposed method.

V. Simulation Result

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

$$y(t) = \frac{\exp(-y^2(t-2)) * y(t-1)}{1+u^2(t-3)+y^2(t-2)} + \frac{\exp(-0.5*(u^2(t-2)+y^2(t-3))) * y(t-2)}{1+u^2(t-2)+y^2(t-1)} + \frac{\sin(u(t-1)*y(t-3))*y(t-3)}{1+u^2(t-1)+y^2(t-3)} + \frac{\sin(u(t-1)*y(t-2))*y(t-4)}{1+u^2(t-2)+y^2(t-2)} + u(t-1). \quad (31)$$

The desired output in this example is a piecewise function:

$$y^*(t) = \begin{cases} 0.4493y^*(t-1)+0.57r(t-1) \\ \qquad t\in[1,100]\cup[151,200] \\ 0.7sign(0.4493y^*(t-1)+0.57r(t-1)) \\ \qquad t\in[101,150] \end{cases}. \quad (32)$$

where $r(t) = 1.2*\text{sign}(2\pi t/25)$.

TABLE I. Comparison Result of Error Each Method

Method	Errors		
	Mean of errors	***Variance***	***MSE***
IQARXNN controller	0.0035	0.0053	0.0052
MENN with MPSO	0.0017	0.0013	0.0013

We use the following improved quasi-ARX neural network prediction model to identify the system:

$$y_I(t+d|t,\varepsilon(t)) = \Psi^T(t)\theta+\chi(t)\ \Psi^T(t)\ .\ W_2\Gamma(W_1\varepsilon(t)+B).(33)$$

In the nonlinear part, a neural network with one hidden layer and 20 hidden nodes is used and other parameters satisfy m=4, n = 3, d = 1. Firstly, the improved quasi-ARX model can be trained off-line by the hierarchical training algorithm as in [19].

Figure 3 shows the performance when the MENN controller with MPSO is used comparison with adaptive controller using switching mechanism based on quasi-ARX neural network.

The parameters of switching law function (19) are choosen c=1.2 and N = 3. In Fig. 3, the dotted line is the desired output, the solid line denotes MENN with MPSO control output, and dashed line shows the IQARXNN control output. Obviously, the control output with the proposed method is nearly consistent with the desired output at most of the time.

The mean of IQARXNN control errors is 0.0035 and the variance is 0.0053. The mean of the proposed method control errors is 0.0017 and the variance is 0.0013. Therefore, our method is better than adaptive control. Table I gives the comparison results of the errors. Obviously, MENN with MPSO controller has better performance than adaptive controller.

Figure 4 shows the switching sequence which 1 is model with nonlinear part and 0 is model without nonlinear part. Even though the model with nonlinear part can often control very well, it degrades sometimes and the model only with linear part has to work until the nonlinear part can recover. Therefore, the linear part will work all the time, but the neural network part will work under the switching sequence.

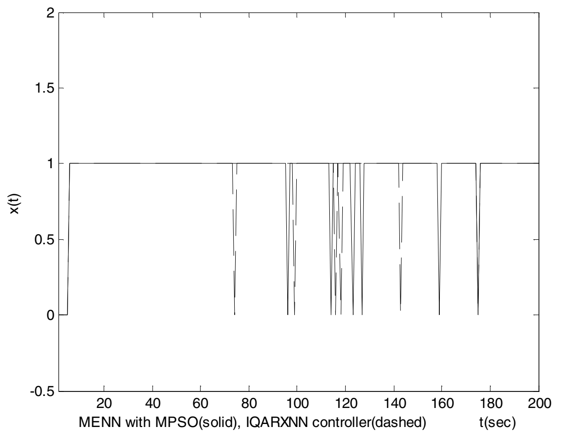

Figure 4. Switching sequence of proposed method.

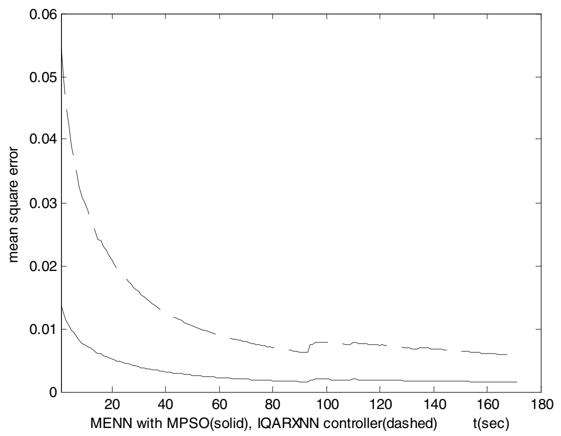

Figure 5. Convergence characteristics of the proposed methods.

TABLE II. CLASSIFICATION RESULT FOR SEVERAL METHODS

Method	*Iterative Number*	*CPU Time (sec)*	*MSE (10^{-3})*	*Accuracy (%)*
IQARXNN controller	273	1.82	5.2	95.87
MENN with MPSO	183	0.56	1.3	98.38

For additional comparisons, convergence speed of the different methods are given in Fig.5, and classified in Table II. From Table II, MENN with MPSO method gets a better accuracy, and also has a faster convergence rate than other methods. Fig. 5 shows the convergent characteristics of various algorithms. A better nonlinear system controller effect can be seen for the proposed algorithm.

VI. CONCLUSION

This study has successfully demonstrated the effectiveness of the proposed modified Elman neural network controller with modified PSO based on quasi-ARX prediction model. First, the principles of quasi-ARX prediction model was derived. Then, the network structure and theoretical bases of proposed modified Elman neural network controller with modified PSO has been adopted to adapt the learning rates in the BP process to replace the traditional trial-and-error method. Finally, the control performance of the proposed modified Elman neural network controller with modified PSO based on quasi-ARX prediction model has been confirmed by some simulated and experimental result.

The main contributions of this study are: (1) the successful development of a modified Elman neural network controller; (2) the successful adoption of a modified PSO algorithm to tune the learning rates in BP process; (3) the successful application of the modified Elman neural network controller with MPSO based on improved quasi-ARX prediction model to control nonlinear system with robust control performance.

ACKNOWLEDGMENT

This research has been supported by Directorate General of Higher Education, Ministry of National Education, (Beasiswa Luar Negeri DIKTI Kementrian Pendidikan dan Kebudayaan Republik Indonesia) and Surabaya Ship Building Polytechnic (Politeknik Perkapalan Negeri Surabaya PPNS).

REFERENCES

[1] Piche, S., Sayyar-Rodsari, B., Johnson, D., Gerules, M., "Nonlinear model predictive control using neural networks", *IEEE Control Systems Magazine",* (3), 53-62, 2000.

[2] Zeng, G. M., Qin, X. S., He, L., Huang, G. H., Liu, H. L., Lin, Y. P., "A neural network predictive control system for paper mill wastewater treatment", *Engineering Application of Artificial Intelligent",* 16, 121-129, 2003.

[3] Wang, S. W., Yu, D. L. Gomm, J. B., Page, G. F., Douglas, S. S., "Adaptive neural network model based predictive control for air-fuel ratio of SI engines", *Engineering Application of Artificial Intelligent",* 19, 189-200, 2006.

[4] Lin, C. T., George Lee, C. S., "Neural Fuzzy Systems", *Prentice Hall*, NJ, USA, 1999.

[5] Wai, R. J., Lin, F. J., "Adaptive recurrent-neural-network control for linear induction motor", *IEEE Transactions on Aerospace and Electronic Systems*, 37 (4), 1176-1192, 2001.

[6] Parlos, A. G., Parthasarathy, A., Atiya, A. F., "Neural-predictive process control using on-line controller adaptation", *IEEE Transactions on Control System Technology*, 9 (5), 741-755, 2001.

[7] Huang J. Q, Lewis, F. L., "Neural-network predictive for nonlinear dynamics systems with time-delay", *IEEE Transactions on Neural Networks,* 14 (2), 377-389, 2003.

[8] Mbede J.B., Huang X., Wang M., "Robust neuro-fuzzy-sensor-based motion control among dynamic obstacles for robot manipulators", *IEEE Trans. Fuzzy Syst.*, 2003, 11, (2), pp. 249-261.

[9] Liu H., Wang S., Ouyang P., "Fault diagnosis based on improved Elman neural network for ahydraulic servo system", *Proc. Int. Conf. Robotics, Automation and Mechatronics*, December 2006, pp. 1-6.

[10] R. H. Middleton and G. C. Goodwin, "Adaptive control of time-varying linear systems," *IEEE Trans. Autom. Control*, vol. AC-33, no. 2, pp. 150–155, Feb. 1988.

[11] A. S. Morse, "Global stability of parameter-adaptive control systems," *IEEE Trans. Autom. Control*, vol. AC-25, no. 3, pp. 433–439, Jun. 1980.

[12] A. S. Morse, "Supervisory control of families of linear set-point controllers-part 1: Exact matching," *IEEE Trans. Autom. Control*, vol. 41, no. 10, pp. 1413–1431, Oct. 1996.

[13] D. R. Mudgett and A. S. Morse, "Adaptive stabilization of linear systems with unknown high-frequency gains," *IEEE Trans. Autom. Control*, vol. AC-30, no. 6, pp. 549–554, Jun. 1985.

[14] F.-J. Lin, L.-T.Teng, H.Chu, "Modified Elman neural network controller with improved particle swarm optimisation for linear synchronous motor drive", *IET Elect. Power Appl.,* 2008, Vol. 2, No. 3, pp. 201-214.

[15] Lin F.J., Wa R.J., Lee C.C, "Fuzzy neural network position controller for ultrasonic motor drive using push-pull DC-DC Converter", *IET Proc. Control Theory Appl.*, 1999, 146, (1), pp. 99-107.

[16] R.Jacob and R. S. Yahya, "Particle swarm optimization in electromagnetics," *IEEE Trans. Antennas Propag.,* vol. 52, no. 2, pp. 397-407, 2004.

[17] W. M. Lin and C. M. Hong, "Intelligent approach to maximum power point tracking control strategy for variable-speed wind turbine generation system," *Energy*, vol. 35, no. 6, pp. 2440-2447, 2010.

[18] W. M. Lin and C. M. Hong, "A new Elman neural network-based control algorithm for adjustable-pitch variabel-speed wind-energy conversion systems," *IEEE Trans. On Power Electronics*, vol. 26, no. 2, pp. 473-481, February, 2011.

[19] J. Hu and K. Hirasawa, "A method for applying neural networks to control of nonlinear systems", *Neural Information Processing: Research and Development*, J. C. Rajapakse and L. Wang, Eds., pp. 351-369, Springer, 5 2004.

[20] I. Sutrisno, "Designing of Rolling Ship FPB-57 Control System with Fins Stabilizer using Self Organizing Corridore Function Adaptif Neuro Fuzzy", *Java Journal of Electrical and Electronics Engineering*, vol. 2, no. 2, ITS, October, 2004.

[21] L. Wang, Y. Cheng and J. Hu, "A Quasi-ARX Neural Network with Switching Mechanism to Adaptive Control of Nonlinear Systems", *SICE Journal of Control, Measurement, and System Integration,* vol. 3, no. 4, pp. 246-252, July, 2010.

[22] 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, July, 2012.

[23] L. Wang, Y. Cheng and J. Hu, " A Quasi-ARX Model for Multivariable Decoupling Control of Nonlinear MIMO System", *Mathematical Problems in Engineering*, vol. 2012, Article ID 570498, 13 pages, doi:10.1155/2012/570498, 2012.