Neurocomputing 135 (2014) 79–85

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

An adaptive PID neural network for complex nonlinear system control

Jun Kang [a,b], Wenjun Meng [a], Ajith Abraham [c,d,e], Hongbo Liu [c,e,*]

[a] School of Mechanical Engineering, Taiyuan University of Science and Technology, Taiyuan 030024, China
[b] Software School, North University of China, Taiyuan 030051, China
[c] Machine Intelligence Research Labs (MIR Labs), Scientific Network for Innovation and Research Excellence, Auburn 98071, WA, USA
[d] IT4Innovations - Center of excellence, VSB - Technical University of Ostrava, Ostrava 70833, Czech Republic
[e] School of Information Science and Technology, Dalian Maritime University, Dalian 116026, China

ARTICLE INFO

Article history:
Received 18 October 2012
Received in revised form
20 February 2013
Accepted 8 March 2013
Available online 4 January 2014

Keywords:
Complex nonlinear system
Adaptive
PID neural network
PSO
Gradient descent

ABSTRACT

Usually it is difficult to solve the control problem of a complex nonlinear system. In this paper, we present an effective control method based on adaptive PID neural network and particle swarm optimization (PSO) algorithm. PSO algorithm is introduced to initialize the neural network for improving the convergent speed and preventing weights trapping into local optima. To adapt the initially uncertain and varying parameters in the control system, we introduce an improved gradient descent method to adjust the network parameters. The stability of our controller is analyzed according to the Lyapunov method. The simulation of complex nonlinear multiple-input and multiple-output (MIMO) system is presented with strong coupling. Empirical results illustrate that the proposed controller can obtain good precision with shorter time compared with the other considered methods.

1. Introduction

In the industrial field, the controlled system usually has great nonlinearity, including spacecraft system, vehicle system, robot system, power system, chemical reaction system, and so on. It is hard to get a precise control performance even by the intelligent control methods, including adaptive control [1,2], fuzzy control [3–5], neural network control [6–8] and decoupling control [9–11]. So many mixed control methods are presented, such as PID neural network. Due to the characteristics of self-learning, self-organizing and self-adaptation, PID neural network would automatically identify the parameters of controlled system and adjust them according to system changes.

In this paper, we present a controller model based on adaptive PID neural networks. To prevent the weights of neural networks falling into local optima, PSO algorithm is adopted to select initial weights. The parameters of PID neural network are self-regulating without intervention. The improved gradient descent method is used to optimize the weights of networks.

2. Related works

Since it is difficult to control a complex nonlinear system [12–14], neural network was introduced to solve the problems. In [15], Jafarnejadsani et al. proposed an adaptive control based on radial-basis-function neural network (NN) for different operation modes of variable-speed variable-pitch wind turbines. In [16], Lin et al. presented an interactively recurrent self-evolving fuzzy neural network to predict and identify the dynamic systems. They derived the consequent update parameters by a variable-dimensional Kalman filter algorithm. In [17], Chemachema introduced a direct adaptive control algorithm based on neural networks for a class of single input single output nonlinear systems. These signals involved in the closed loop were proven to be exponentially bounded and hence the system stability, without any additional control term to the NN adaptive controller. However, researches still confront some difficulties. For example, network parameter training is time-consuming and easily falls into local minimum. Particle swarm optimization (PSO) algorithm is a new globe optimization algorithm, which has the advantage of fast convergence speed [18,19]. In [20], Selvakumaran et al. proposed a new design of decentralized load-frequency controller for interconnected power systems with ac–dc parallel using PSO algorithm. The experiment result illustrated that their method have rapid dynamic response ability. In [21], Hasni et al. used PSO algorithm to parameters selection, and used genetic algorithm to optimize the choice of parameters by minimizing a cost function. The study was applied to a greenhouse environment with Continuous Roof Vents, and obtained satisfactory effect. Nevertheless, it is difficult to apply directly these methods to complex nonlinear system with strong coupling.

Adaptive controller has the ability to adjust of control parameters without the help of human intelligence. It can tune complex systems better by combining nonlinear controlling methods and intelligent control technology [22,23]. The results show that adaptive control has the advantage to solve effectively the

* Corresponding author.
E-mail addresses: kj_ty0015@sina.com (J. Kang), tyustmwj@126.com (W. Meng), ajith.abraham@ieee.org (A. Abraham), lhb@dlut.edu.cn (H. Liu).

0925-2312/$ - see front matter © 2014 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2013.03.065

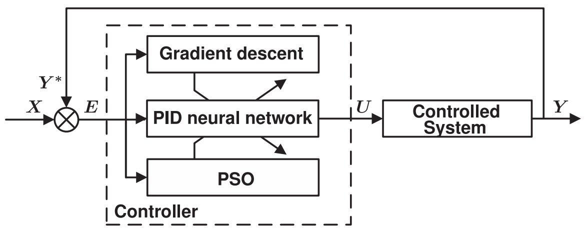


Fig. 1. Structure of control system.

problems of nonlinear system with uncertain model and random disturbance.

3. Adaptive PID neural networks

3.1. Control system structure

The control system adopts close loop control, and it mainly consists of two parts: the controller and the controlled system, as shown in Fig. 1. The controller is built based on adaptive PID neural network. In the whole control system, X is object vector, E is error vector, Y is output value of control system and U is control law of the control system. The controlling algorithm is illustrated in Algorithm 1.

Algorithm 1. Controlling algorithm for complex nonlinear system.

```
1: Input the object value of controlled system into the
   controller.
2: Initialize weights of PID neutral network by PSO algorithm.
3: Use PID neural network to control the controlled system.
4: Feedback the output of the control system.
5: Adjust parameters of PID neural network by improved
   gradient descent method.
6: If the control error is small enough, algorithm is terminated.
   If not, return to Step 3.
```

3.2. PID neural network controller

In the controller, three-layer PID neural network is built by combining PID and feedforward neural network, as shown in Fig. 2. $X^* = [XY]$ is input vector of the controller, $X = [x_1, x_2, \ldots, x_n]^T$ is object value of the whole control system, and $Y = [y_1^*, y_2^*, \ldots, y_n^*]^T$ is a feedback value from current system output.

Input layer has $2n$ neurons, n of them are used to input object values, the others are used to input values which returned from control system's output. The output of this layer at k is

$$out_{q1}^1(k) = x_q(k) \tag{1}$$

$$out_{q2}^1(k) = y_q^*(k) \tag{2}$$

Hidden layer has $3n$ neurons, including n proportion neurons, n integration neurons and n differentiation neurons. The output of each neuron in this layer at k is

$$out_{q2}^2(k) = \phi_p \sum_{i=1}^{2} \omega_{i1} out_{qi}^1(k) \tag{3}$$

$$out_{q2}^2(k) = \phi_i \left[\sum_{i=1}^{2} \omega_{i2}(k) x_{li}(k) + out_{q2}^2(k-1) \right] \tag{4}$$

$$out_{q3}^2(k) = \phi_d \left[\sum_{i=1}^{2} \omega_{i3}(k) x_{li}(k) - \sum_{i=1}^{2} \omega_{i3}(k-1) x_{li}(k-1) \right] \tag{5}$$

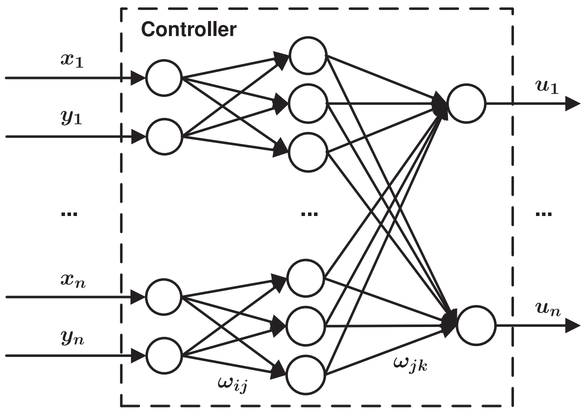


Fig. 2. Structure of PID neural network.

where, ϕ_p, ϕ_i and ϕ_d are coefficient, usually larger than 1, which is used to balance output values from proportion neurons, integration neurons and differentiation neurons. Output layer has n neurons. The output of each neuron in this layer at k is

$$u_p(k) = out_p^3(k) = \sum_{l=1}^{n} \sum_{j=1}^{3} \omega_{jp}(k-1) out_{qj}^2(k-1) \tag{6}$$

where q is the number of subnets, that is, the number of output values. And j is the number of neurons in hidden layer, ω_{ij} is the weight between input layer and hidden layer, ω_{jk} is the weight between hidden layer and output layer.

3.3. Parameters initiation

PSO algorithm searches for the optimal solution by collaboration among individuals in the population [24,25]. In the algorithm, weight initiation is done randomly. However, weights may fall into local optima during the process of optimization. In this paper, PSO algorithm is adopted to set initial weights in the controller. The main steps of PSO algorithm are showed in Algorithm 2.

Algorithm 2. Particle swarm optimization algorithm.

```
1: Initialize a group of individuals by random algorithm
   (population size is m), including random position and
   velocity.
2: Calculate the fitness of each individual.
3: For each individual, compare the fitness with the fitness of
   its best historical position bhp. If the former is superior to
   the latter, bhp will be replaced with the current fitness, and
   the position of bhp will also be replaced with the current
   position.
4: For each individual, compare the fitness with the fitness of
   global best historical position gbhp. When the former is
   superior to the latter, gbhp will be replaced with the
   subscript and fitness of current individual.
5: Update the position and velocity of particles.
6: Check end condition. If satisfied, algorithm is over,
   otherwise, k=k+1, return to Step 2. The end condition is
   that a good enough fitness or the max desired evolution
   population mdep reaches.
```

In Step 5, the position and velocity of particles are updated according to the following equations:

$$v_{id}^{k+1} = v_{id}^k + a\psi_1 (p_{id}^k - x_{id}^k) + b\psi_2 (p_{gd}^k - x_{id}^k) \tag{7}$$

$$x_{id}^{k+1} = x_{id}^k + v_{id}^{k+1} \tag{8}$$

where $d \in [1, 2, \ldots, n]$, $i \in [1, 2, \ldots, m]$, k is current evolution population, ψ_1 and ψ_2 are random number between 0 and 1, a and b are acceleration constants. In order to prevent velocity of individual against great change, a max velocity is limited to a maximum of $V_{\max}$.

3.4. Adaptive parameters adjustment

Usually to get better control effect and close quickly to control object values, weights must be adjusted according to the error. The gradient descent method can be used to adjust the velocity. For function $f(X)$, $X = (x_1, x_2, \ldots, x_n)$, the gradient is shown as follows:

$$\operatorname{grad} f(X) = \left[\frac{\partial f(X)}{\partial x_1}, \frac{\partial f(X)}{\partial x_2}, \ldots, \frac{\partial f(X)}{\partial x_n}\right]^T \tag{9}$$

Here the minus gradient direction is the steepest descent direction. Gradient information is added into individual velocity with certain probability, which help particle to search the solutions more efficiently. The weights are updated by Eqs. (10) and (11):

$$\omega_{ij}(k+1) = \omega_{ij}(k) - \mu \frac{\partial e(k)}{\partial \omega_{ij}(k)} + \sigma[\omega_{ij}(k) - \omega_{ij}(k-1)] \tag{10}$$

$$\omega_{jp}(k+1) = \omega_{jp}(k) - \mu \frac{\partial e(k)}{\partial \omega_{jp}(k)} + \sigma[\omega_{jp}(k) - \omega_{jp}(k-1)] \tag{11}$$

where μ and σ are the network learning rates, and $e(k)$ is the control error calculated as follows:

$$e(k) = \frac{1}{2} \sum_{q=1}^{n} |y_q^*(k) - x_q(k)|^2 \tag{12}$$

In our neural network, the parameters are computed in each sampling period. The weights are automatically adjusted according to the errors of closed loop system, and the controller helps us to implement nonlinear and adaptive real-time online control for controlled system.

4. Stability analysis

Let Lyapunov function be

$$V(k) = \frac{1}{2} \sum_{q=1}^{n} e_0^2(k) \tag{13}$$

where

$$e_0(k) = y_q^*(k) - x_q(k) \tag{14}$$

The change of Lyapunov function is

$$\Delta V(k) = V(k+1) - V(k) = \frac{1}{2} \sum_{q=1}^{n} ((e_0^2(k+1) - e_0^2(k))) \tag{15}$$

That is

$$\Delta V(k) = \frac{1}{2} \sum_{q=1}^{n} ((e_0(k+1) - e_0(k)))(e_0(k+1) + e_0(k)) = \frac{1}{2} \sum_{q=1}^{n} \Delta e_0(k)(2e_0(k) + \Delta e_0(k)) = \sum_{q=1}^{n} e_0(k)\Delta e_0(k) + \frac{1}{2} \sum_{q=1}^{n} \Delta e_0^2(k) \tag{16}$$

According to the Lyapunov Theorem [26], the closed loop system is stable if $\Delta V(k) \le 0$ in any sampling period. That is,

$$\sum_{q=1}^{n} e_0(k)\Delta e_0(k) \le \frac{1}{2} \sum_{q=1}^{n} \Delta e_0^2(k) \tag{17}$$

Based on Eq. (10) to Eq. (12), we can obtain as follows:

$$\Delta \omega_{ij}(k) = -\frac{\mu}{1-\sigma} \frac{\partial e(k)}{\partial \omega_{ij}(k)} \approx -\frac{\mu}{1-\sigma} e_0(k) \left. \sum_{q=1}^{n} e_0(k) \operatorname{sgn} \frac{\Delta y_q^*(k)}{\Delta x_q(k)} \right) \Big) \tag{18}$$

Then

$$\Delta e_0(k) = \sum_{q=1}^{n} \sum_{i=1}^{3} \left(\frac{\partial e_0(k)}{\partial \omega_{ij}(k)} \Delta \omega_{ij}(k) \right) \Big) = -\frac{\mu}{1-\sigma} \left. \sum_{i=1}^{3} \frac{1}{\omega_{ij}(k)} \right) \sum_{q=1}^{n} \left(e_0(k) \operatorname{sgn}\left(\frac{\Delta y_q(k)}{\Delta x_q(k)} \right) \right) \sum_{q=1}^{n} e_0^2(k) \tag{19}$$

Let $h(k) = \sum_{q=1}^{n} (e_0(k) \operatorname{sgn}(\Delta y_q(k)/\Delta x_q(k))) \sum_{q=1}^{n} e_0^2(k)$, Eq. (19) is simplified to

$$\Delta e_0(k) = \sum_{q=1}^{n} \sum_{i=1}^{3} \left(\frac{\partial e_0(k)}{\partial \omega_{ij}(k)} \Delta \omega_{ij}(k) \right) \Big) = -\frac{\mu}{1-\sigma} \left. \sum_{i=1}^{3} \frac{1}{\omega_{ij}(k)} \right) h(k) \tag{20}$$

Substitute to Eq. (17), then

$$-\frac{\mu}{1-\sigma} h(k) \sum_{q=1}^{n} e_0(k) \left. \sum_{i=1}^{3} \frac{1}{\omega_{ij}(k)} \right) \le \frac{1}{2} \left(\frac{\mu}{1-\sigma} \right)^2 h^2(k) \sum_{q=1}^{n} \left. \sum_{i=1}^{3} \frac{1}{\omega_{ij}(k)} \right)^2 \tag{21}$$

To ensure $\Delta V(k) \le 0$, consider two cases as follows:

1. When $(\sum_{q=1}^{n} \sum_{i=1}^{3} e_0(k)/\omega_{ij}(k))h(k) < 0$, then

$$0 < \frac{\mu}{\sigma - 1} \le -2 \frac{\sum_{q=1}^{n} \sum_{i=1}^{3} \frac{e_0(k)}{\omega_{ij}(k)}}{\left(\sum_{q=1}^{n} \left(\sum_{i=1}^{3} \frac{1}{\omega_{ij}(k)} \right)^2 \right) h(k)} \tag{22}$$

2. When $(\sum_{q=1}^{n} \sum_{i=1}^{3} e_0(k)/\omega_{ij}(k))h(k) \ge 0$, then

$$-2 \frac{\sum_{q=1}^{n} \sum_{i=1}^{3} \frac{e_0(k)}{\omega_{ij}(k)}}{\left(\sum_{q=1}^{n} \left(\sum_{i=1}^{3} \frac{1}{\omega_{ij}(k)} \right)^2 \right) h(k)} \le \frac{\mu}{\sigma - 1} \le 0 \tag{23}$$

5. Simulation

A simulation is carried out to verify the proposed control strategy in this section. The controlled system is a complex nonlinear multiple-input and multiple-output (MIMO) system with strong coupling of variables, which is determined as follows:

$$\begin{cases} y_1(k+1) = 0.4 \times y_1(k) + 0.3 \times y_2(k) + \dfrac{u_1(k)}{1 + u_1(k)^2} + 0.2 \times u_2(k)^3 \\ y_2(k+1) = 0.3 \times y_2(k) + 0.2 \times y_3(k) + \dfrac{u_2(k)}{1 + u_2(k)^2} + 0.6 \times u_1(k)^3 \\ y_3(k+1) = 0.5 \times y_3(k) + 0.3 \times y_1(k) + \dfrac{u_3(k)}{1 + u_3(k)^2} + 0.1 \times u_2(k)^3 \end{cases} \tag{24}$$

where u_1, u_2, u_3 are control laws. The following parameters are set to the system:

- Initial values of control system are specified as [0 00].
- Object values of control system are specified as [0.7 0.4 0.8].

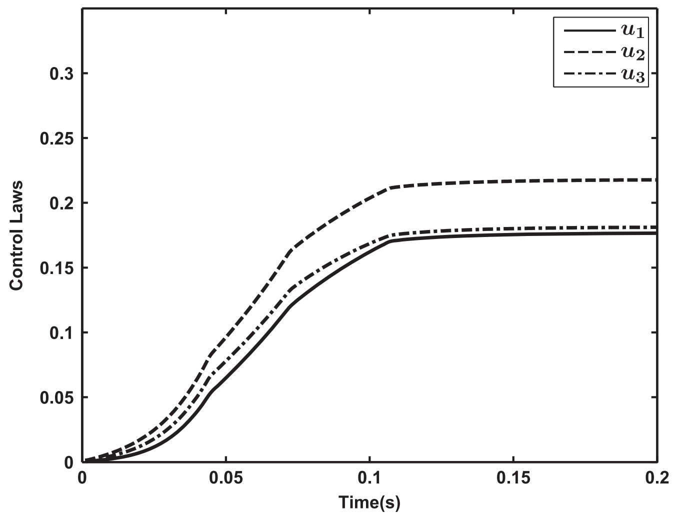


Fig. 3. Control laws vary with time.

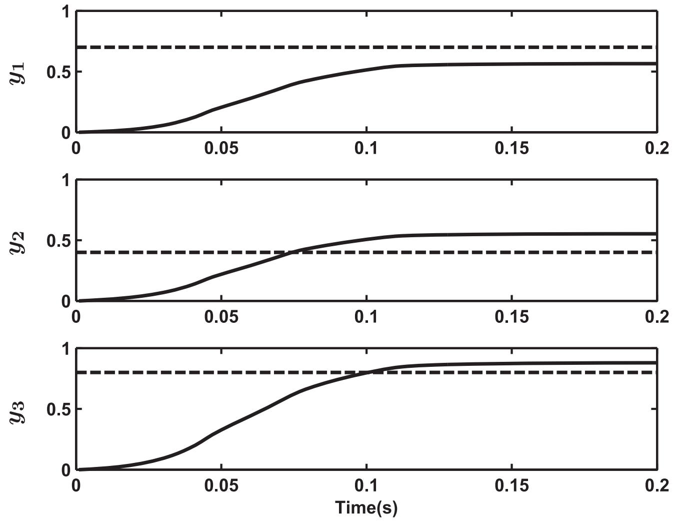


Fig. 4. Actual output values vary with time.

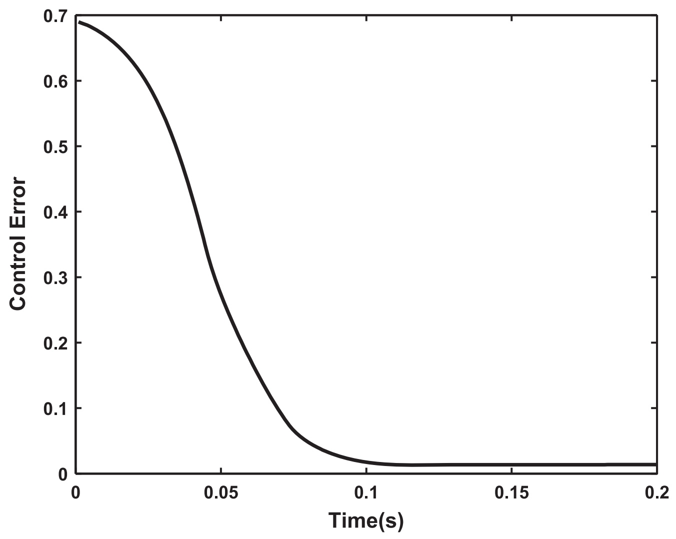


Fig. 5. Control error varies with time.

- Learning rates is specified as 0.006.
- Time interval is specified as 0.0001 s.

During the process of weights initiation by PSO, numbers of populations is specified as 50, and iteration number is specified as 40. To illustrate the advantages of our controller, three different methods are used to control the same system separately.

5.1. *Control by traditional PID neutral network*

The first simulation is controlling the system by traditional PID neutral network. Fig. 3 illustrates the control laws changed with time. Fig. 4 shows the contrast between the actual output values and the object output values. Fig. 5 shows control error with time variation.

Simulation results show that the actual output is close to expected output, control law is gradually stabilized, and control error is close to 0. That is to say, this method has some effect on control the system.

5.2. *Control by PID neutral network though standard PSO optimization*

The second simulation is controlling the same system by PID neutral network which is optimized by standard PSO. Fig. 6 shows the control laws changed with time. Fig. 7 depicts the contrast between the actual output values and the object output values. Fig. 8 illustrates the control error with time variation.

Simulation results show that the actual output is close to the expected output, and the speed of convergence is faster than the previous method and the control law is gradually stabilized.

5.3. *Control by the adaptive PID neutral network*

The last simulation is controlling the same system by adaptive PID neutral network, which is proposed in this paper. The empirical results are shown from Figs. 9 to 11. Fig. 9 shows the control laws changed with time. Fig. 10 shows the contrast between the actual output values and the object output values. Fig. 11 shows control error with time variation.

In order to compare the performance of three different control methods mentioned above, the control errors varied with time are shown in Table 1. I, II and III denote the control error by 4.1, 4.2, 4.3 respectively. Its time interval of data is 0.001 s, and 15 data groups are selected from 0.02 s. As the results illustrated, the actual output

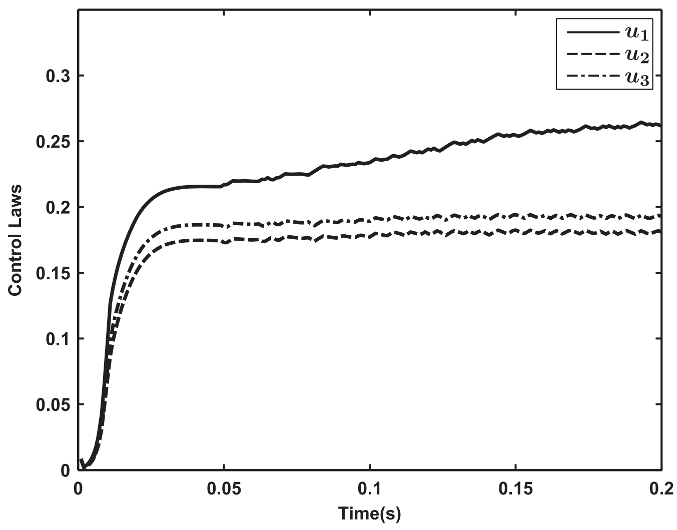


Fig. 6. Control laws vary with time.

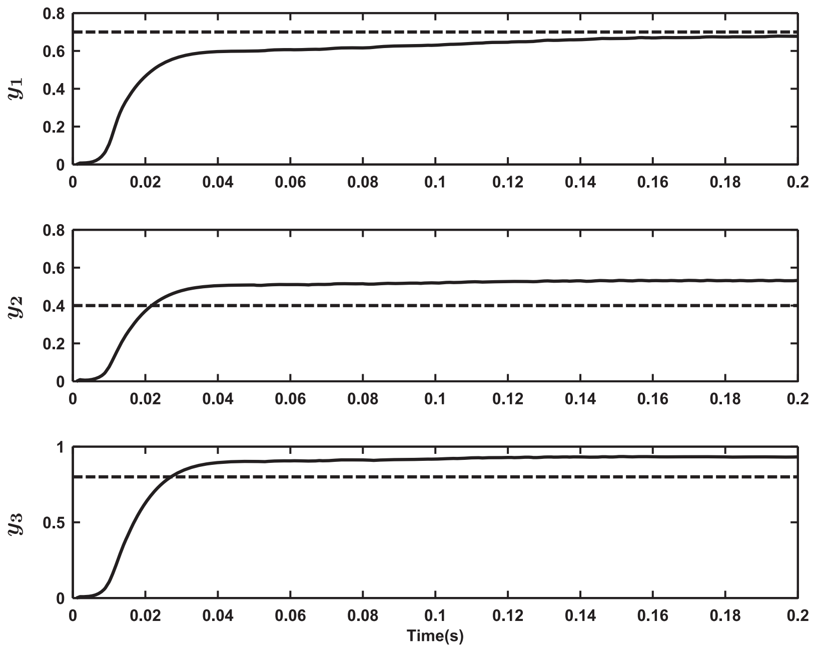


Fig. 7. Actual output values vary with time.

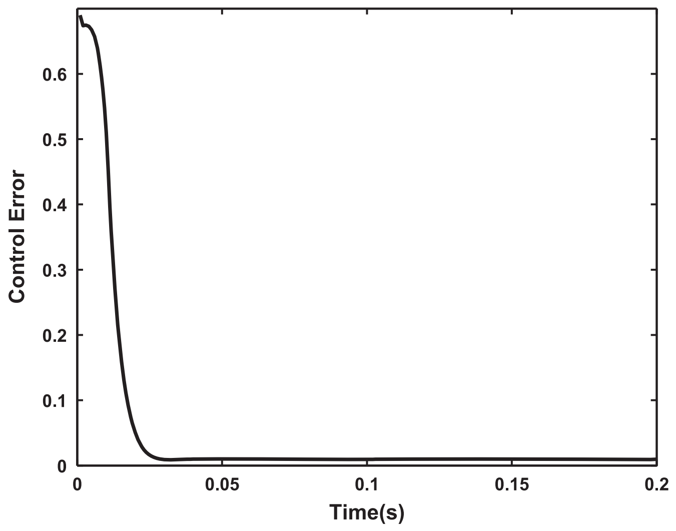


Fig. 8. Control error varies with time.

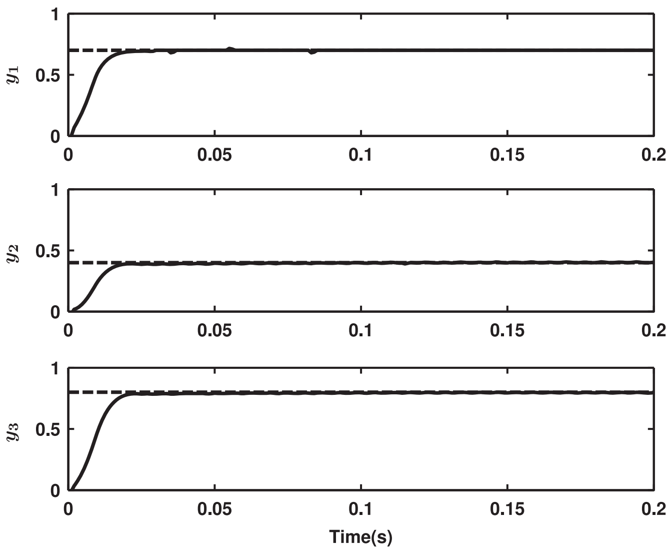


Fig. 10. Actual output values vary with time.

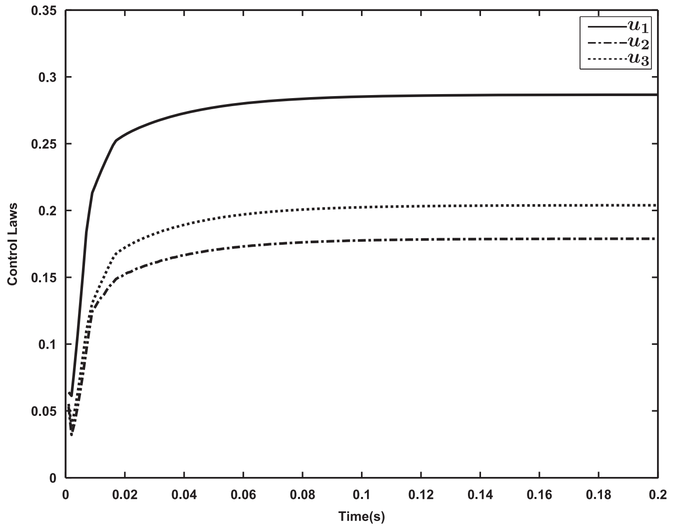


Fig. 9. Control laws vary with time.

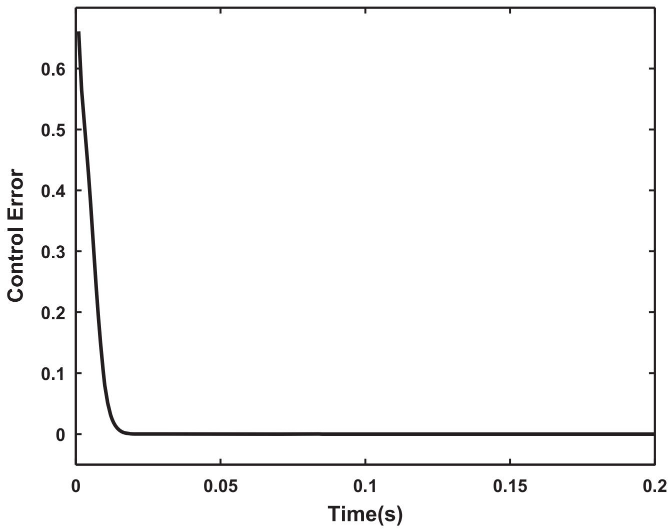


Fig. 11. Control error varies with time.

Table 1
Control error varies with time using above three control methods.

Groups	Times	I	II	III
1	0.020	0.626280	0.050725	0.000322
2	0.021	0.620422	0.039564	0.000204
3	0.022	0.614219	0.030990	0.000174
4	0.023	0.607665	0.024495	0.000165
5	0.024	0.600709	0.019648	0.000168
6	0.025	0.593351	0.016086	0.000221
7	0.026	0.585584	0.013522	0.000194
8	0.027	0.577358	0.011724	0.000158
9	0.028	0.568675	0.010503	0.000185
10	0.029	0.559526	0.009712	0.000234
11	0.030	0.549869	0.009237	0.000214
12	0.031	0.539705	0.008988	0.000165
13	0.032	0.529032	0.008897	0.000103
14	0.033	0.517811	0.008912	0.000103
15	0.034	0.506049	0.008997	0.000138
Mean	–	0.573084	0.018133	0.000183

values can quickly approximate the object output values by our proposed method. The control error is falling faster before 0.02 s, then tends to 0 gradually. The control laws are also quickly changed to constant within a short time. Therefore, the adaptive PID neutral network controller has fast convergence speed, high accuracy and reliable stability for the complex nonlinear systems.

6. Conclusion

In this paper, an adaptive PID neural network controller was presented, which model is constructed based on a PID neural network. PSO algorithm was adopted to select initial weights in its training for improving the convergent speed and preventing the weights getting trapped into local optima. In each sampling period, an improved gradient descent method was used to update the weights in this network. Its adaptive parameter adjustment features self-correcting, on-line and real-time. The stability of our controller is analyzed according to the Lyapunov method.

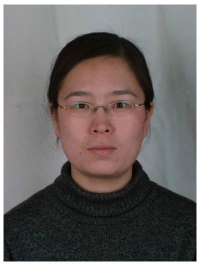
Empirical results illustrate that the adaptive PID neural network controller is significantly better than the other considered methods. Our controller can achieve better control results within less sampling periods and the error tends to 0 in a stable manner. During the weight initialization, PSO algorithm takes a long time, which requires more research works on how to decide the number of iterations for a better balance between its efficiency and precision.

Acknowledgments

This work is supported partly by the National Natural Science Foundation of China (Grant Nos. 51075289, 61173035), the Shanxi Natural Science Foundation (Grant No. 2011011019-3), the project GACR P202/10/0262, the IT4Innovations Centre of Excellence project, reg. no. CZ.1.05/1.1.00/02.0070 supported by Operational Programme 'Research and Development for Innovations' funded by Structural Funds of the European Union and state budget of the Czech Republic and by long-term strategic development financing of the Institute of Computer Science (RVO: 67985807).

References

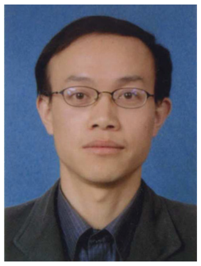
[1] A. Boulkroune, M. Tadjine, M. M'Saad, M. Farza, Fuzzy adaptive controller for MIMO nonlinear systems with known and unknown control direction, Fuzzy Sets Syst. 161 (6) (2010) 797–820.
[2] N. Reza, R. Ahmad, Adaptive decoupled control of 4-leg voltage-source inverters for standalone photovoltaic systems: adjusting transient state response, Renew. Energy 36 (10) (2011) 2733–2741.
[3] J. Liu, W. Wang, F. Golnaraghi, E. Kubica, Fuzzy adaptive observer backstepping control for MIMO nonlinear systems, Fuzzy Sets Syst. 160 (21) (2010) 2746–2759.
[4] Y. Wang, L. Nie, Application of fuzzy control on smart car servo steering system, Adv. Sci. Lett. 4 (6–7) (2011) 2099–2103.
[5] D. Rastovic, Tokamak design as one sustainable system, Neural Netw. World 21 (6) (2011) 493–504.
[6] M. Chen, C. Jiang, Q. Wu, Disturbance-observer-based robust flight control for hypersonic vehicles using neural networks, Adv. Sci. Lett. 4 (4–5) (2011) 1771–1775.
[7] V. Akpan, G. Hassapis, Nonlinear model identification and adaptive model predictive control using neural networks, ISA Trans. 50 (2) (2011) 177–194.
[8] E. Irigoyen, M. Larrea, J. Valera, V. Gómez, F. Artaza, A hybridized neuro-genetic solution for controlling industrial r3 workspace, Neural Netw. World 20 (7) (2010) 811–824.
[9] J. Garrido, F. Vžquez, F. Morilla, Centralized multivariable control by simplified decoupling, J. Process Control 22 (6) (2012) 1044–1062.
[10] C. Cha, S. Kim, L. Cao, H. Kong, Decoupled control of stiffness and permeability with a cell-encapsulating poly (ethylene glycol) dimethacrylate hydrogel, Biomaterials 31 (18) (2010) 4864–4871.
[11] I. García-Herreros, X. Kestelyn, J. Gomand, R. Coleman, P. Barre, Model-based decoupling control method for dual-drive gantry stages: a case study with experimental validations, Control Eng. Pract. 21 (3) (2013) 298–307.
[12] A. Nassirharand, Computer-aided Nonlinear Control System Design, Springer Verlag, 2012.
[13] L. Grigsby, Power System Stability and Control, third edition, CRC Press, 2012.
[14] A. Veloni, A. Palamides, Control System Problems: Formulas, Solutions, and Simulation Tools, CRC Press, 2012.
[15] H. Jafarnejadsani, J. Pieper, J. Ehlers, Adaptive control of a variable-speed variable-pitch wind turbine using radial-basis function neural network, IEEE Trans. Control Syst. Technol. 34 (2013) 34. (Online).
[16] Y.-Y. Lin, J.-Y. Chang, C.-T. Lin, Identification and prediction of dynamic systems using an interactively recurrent self-evolving fuzzy neural network, IEEE Trans. Neural Netw. Learn. Syst. 24 (2) (2013) 310–321.
[17] M. Chemachema, Output feedback direct adaptive neural network control for uncertain SISO nonlinear systems using a fuzzy estimator of the control error, Neural Netw. 26 (2012) 25–34.
[18] A. Abraham, H. Guo, H. Liu, Swarm intelligence: foundations, perspectives and applications, Swarm Intell. Syst. 26 (2006) 3–25.
[19] Y.-S. Yang, W.-D. Chang, T.-L. Liao, Volterra system-based neural network modeling by particle swarm optimization approach, Neurocomputing 82 (2012) 179–185.
[20] S. Selvakumaran, S. Parthasarathy, R. Karthigaivel, V. Rajasekaran, Optimal decentralized load frequency control in a parallel AC–DC interconnected power system through HVDC link using PSO algorithm, Energy Procedia 14 (2012) 1849–1854.
[21] A. Hasni, B. Draoui, M. Latfaoui, T. Boulard, Identification of natural ventilation parameters in a greenhouse with continuous roof vents, using a PSO and GAs, Sensors Transducers 119 (8) (2010) 182–192.
[22] A. Boulkroune, M. M'Saad, M. Farza, Adaptive fuzzy tracking control for a class of MIMO nonaffine uncertain systems, Neurocomputing 93 (15) (2012) 48–55.
[23] H. Wang, B. Chen, C. Lin, Adaptive neural control for strict-feedback stochastic nonlinear systems with time-delay, Neurocomputing 77 (1) (2012) 267–274.
[24] D. Chen, C. Zhao, Particle swarm optimization with adaptive population size and its application, Appl. Soft Comput. 9 (1) (2009) 39–48.
[25] S. Klein, J. Pluim, M. Staring, M. Viergever, Adaptive stochastic gradient descent optimisation for image registration, Int. J. Comput. Vis. 81 (3) (2009) 227–239.
[26] A. Lyapunov, Stability of Motion, Academic Press, New-York and London, 1966.



Jun Kang received her B.Eng. degree in computer and its application in 2000, M. Eng. degree in artificial intelligence in 2005, both from North University of China, China. Currently, she is a Ph.D. candidate at Taiyuan University of Science and Technology, China. Her research interests include mechanical and electrical system control, and artificial intelligence.

Wenjun Meng received his B.Eng. degree in lifting and conveying mechanism in 1984 and M. Eng. degree in construction mechanism in 1990, both from Taiyuan University of Science and Technology (TYUST), China. He received his Ph.D. degree in mechanical and electronics engineering from Beijing University of Aeronautics & Astronautics), China, in 2005. Currently, he is a professor and dean with the School of Mechanical Engineering, TYUST. His research interests include all aspects of Study and Control for Mechanical and Electrical System, the fundamentals and industrial applications of solids handling and storage systems including mechanical conveying, hoppers and control, material quality, characterization and instrumentation, and so on.

Ajith Abraham is the Director of Machine Intelligence Research Labs (MIR Labs), Scientific Network for Innovation and Research Excellence, WA, USA, with a joint appointment in the IT For Innovations - Center of excellence at VSB - Technical University of Ostrava, Czech Republic. He also holds an Adjunct Professor appointment in Dalian Maritime University, China. He received the Ph.D. degree in Computer Science from Monash University, Melbourne, Australia. His research and development experience includes more than 23 years in the industry and academia. He works in a multidisciplinary environment involving machine intelligence, network security, various aspects of networks, e-commerce, Web intelligence, Web services, computational grids, data mining and their applications to various real-world problems. He has authored/co-authored more than 900 publications (h-index=50+), and some of the works have also won best paper awards at international conferences. He has given more than 60 plenary lectures and conference tutorials in these areas. He serves/has served the editorial board of over 50 International journals and has also guest edited over 40 special issues on various topics. Since 2008, he is the Chair of IEEE Systems Man and Cybernetics Society Technical Committee on Soft Computing and a Distinguished Lecturer of IEEE Computer Society representing Europe (since 2011). More information at: http://www.softcomputing.net.



Hongbo Liu received his three level educations (B.Sc., M.Sc., Ph.D.) at the Dalian University of Technology, China. Currently he is a professor at School of Information Engineering and Science, Dalian Maritime University, with an affiliate appointment in the Institute for Neural Computation, University of California San Diego, USA. His research interests are in system modeling and optimization involving soft computing, probabilistic modeling, cognitive computing, machine learning, data mining, and so on. He participates and organizes actively international conference and workshop and international journals/publications.