Available online at www.sciencedirect.com
SCIENCE DIRECT®
Applied Soft Computing xxx (2006) xxx–xxx
Applied Soft Computing
www.elsevier.com/locate/asoc

Modeling and control of non-linear systems using soft computing techniques

M.A. Denaï [a,*], F. Palis [b], A. Zeghbib [b]

[a] Faculty of Electrical Engineering, USTO, BP 1505 El-Mnaouar, Oran 31000, Algeria
[b] Institut für Elektrische Energiesysteme, Universität Magdeburg, Postfach 4120, 39016 Magdeburg, Germany

Received 12 April 2004; received in revised form 14 December 2005; accepted 18 December 2005

Abstract

This work is an attempt to illustrate the utility and effectiveness of soft computing approaches in handling the modeling and control of complex systems. Soft computing research is concerned with the integration of artificial intelligent tools (neural networks, fuzzy technology, evolutionary algorithms, . . .) in a complementary hybrid framework for solving real world problems. There are several approaches to integrate neural networks and fuzzy logic to form a neuro-fuzzy system. The present work will concentrate on the pioneering neuro-fuzzy system, Adaptive Neuro-Fuzzy Inference System (ANFIS). ANFIS is first used to model non-linear knee-joint dynamics from recorded clinical data. The established model is then used to predict the behavior of the underlying system and for the design and evaluation of various intelligent control strategies.

Keywords: ANFIS; Neural networks; Modeling; Intelligent control

1. Introduction

The current trend in intelligent systems or soft computing research is concerned with the integration of artificial intelligent tools (neural networks, fuzzy technology, evolutionary algorithms, . . .) in a complementary hybrid framework for solving complex problems.

Fuzzy logic offers the important concept of fuzzy set theory, fuzzy if-then rules and approximate reasoning which deals with imprecision and information granularity. Neural networks have on their side the capability for learning and adaptation by adjusting the interconnections between layers, while genetic algorithms make use of a systemized random search and are important for optimization.

Neuro-fuzzy techniques have emerged from the fusion of Artificial Neural Networks (ANN) and Fuzzy Inference Systems (FIS) and form a popular framework for solving real world problems. A neuro-fuzzy system is based on a fuzzy system which is trained by a learning algorithm derived from neural network theory. While the learning capability is an advantage from the viewpoint of FIS, the formation of linguistic rule base will be advantageous from the viewpoint of ANN. There are several approaches to integrate ANN and FIS and very often the choice depends on the application [1,2].

The growing interest in the field is demonstrated by the ever-increasing applications in various areas extending from image and pattern recognition to identification and control applications.

Intelligent control emerged as a viable alternative to conventional model-based control schemes. This is because with fuzzy logic and neural networks issues such as uncertainty or unknown variations in plant parameters and structure can be dealt with more effectively and hence improving the robustness of the control system.

The present work will concentrate on the pioneering neuro-fuzzy system, Adaptive Neuro-Fuzzy Inference System (ANFIS) [3], which is presently available in MatLab®. ANFIS belongs to the class of rules extracting systems using a decompositional strategy, where rules are extracted at the level of individual nodes within the neural network and then aggregating these rules to form global behavior descriptions. The objective of this research work is to explore a number of control strategies for Functional Electrical Stimulation (FES) induced gait.

FES is a rehabilitative technology that can restore muscle activity to people who have suffered spinal cord injury and become paralyzed. The technique consists of applying a variable pulsewidth input signal in order to alter the level of

* Corresponding author. Tel.: +44 114 222 5613; fax: +44 114 222 5661.
E-mail address: m.denai@sheffield.ac.uk (M.A. Denaï).

1568-4946/$ – see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2005.12.005

contraction of the quadriceps muscle group to perform the motion of the shank. The output signal of the system is the angle between the thigh and shank.

High performance FES control system design relies heavily on the availability of a precise knee-joint model. ANFIS like systems have proven to be able to deliver very accurate models and are therefore suitable candidates for such applications [4].

The non-linear, time-variant behavior of the knee-joint dynamics under FES has led us to investigate various control strategies based on neural networks and adaptive neuro-fuzzy approaches.

It has been shown that with the aid of ANFIS, it is possible to create a Fuzzy Inference System that emulates the behavior of the neuromuscular system on the basis of available recorded real-time data.

For controller design and performance evaluation purposes this ANFIS based knee-joint model will be is used in the simulation study.

2. Neuro-fuzzy modeling approach

Conventional approaches to system modeling rely heavily on mathematical tools which emphasizes a precise description of the physical quantities involved. By contrast, modeling approaches based on neural networks and fuzzy logic are becoming a viable alternative where the former conventional techniques fail to achieve satisfactory results.

Neuro-fuzzy modeling is concerned with the extraction of models from numerical data representing the behavioral dynamics of a system.

This modeling approach has a two-fold purpose:

- It provides a model that can be used to predict the behavior of the underlying system.
- This model may be used for controller design.

The main steps of a fuzzy inference, i.e. fuzzyfication of the input physical variables and computation of the degree of satisfaction of the available linguistic terms, the conjunction of the premises as well as the actual fuzzy inference and defuzzyfication, are realized in sequentially ordered layers of a neural network with an architecture such that the weights to be adjusted in the network (usually by means of a gradient descent algorithm) have a meaning as parameters of the rules to be extracted.

2.1. Overview of ANFIS

ANFIS implements a Takagi Sugeno FIS and has a five-layered architecture as shown in Fig. 1.

- The first layer represents fuzzy membership functions.
- The second and the third layers contain nodes that form the antecedent parts in each rule.
- The fourth layer calculates the first-order Takagi-Sugeno rules for each fuzzy rule.
- The fifth layer—the output layer, calculates the weighted global output of the system.

ANFIS uses backpropagation learning to determine premise parameters and least mean squares estimation to determine the consequent parameters. This is referred to as hybrid learning. A step in the learning procedure has got two passes: in the first or forward pass, the input patterns are propagated, and the optimal consequent parameters are estimated by an iterative least mean square procedure, while the premise parameters are assumed to be fixed for the current cycle through the training set. In the second or backward pass the patterns are propagated again, and in this epoch, back propagation is used to modify the premise parameters, while the consequent parameters remain fixed. This procedure is then iterated until the error criterion is satisfied. A detailed description on ANFIS architecture and learning procedure is given in Appendix A.

2.2. ANFIS based modeling of knee-joint dynamics

Given the real time input–output data of the knee-joint system, ANFIS as part of MatLab's Fuzzy Logic Toolbox has

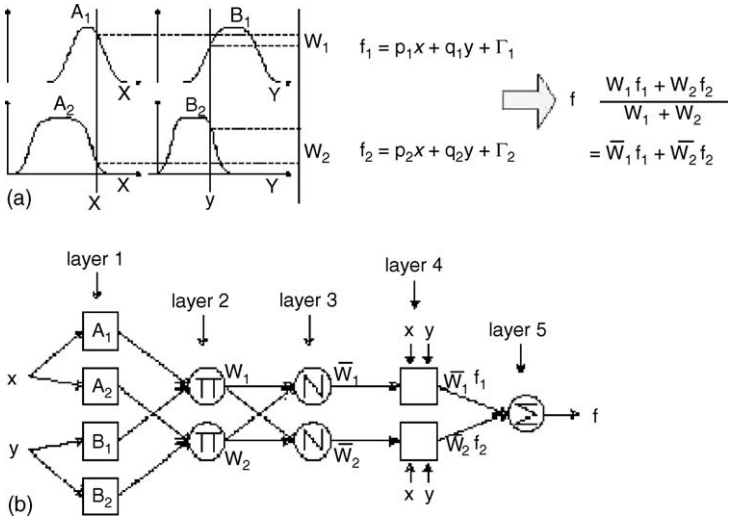

Fig. 1. (a) A two-input first-order Sugeno fuzzy model and (b) equivalent ANFIS architecture for two rules.

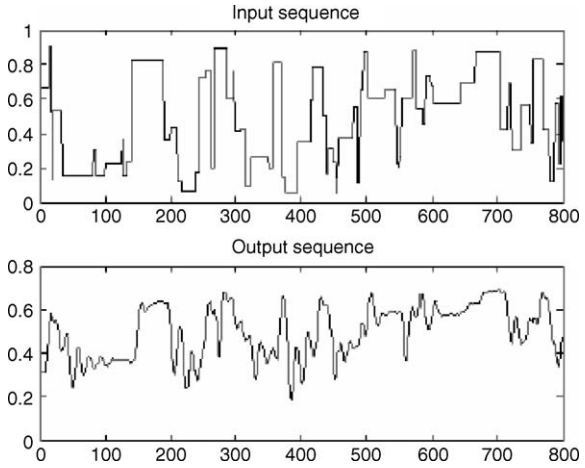

Fig. 2. Real time data set: electrical stimuli (top) and measured neuromuscular response (bottom).

been used effectively for building a set of fuzzy if-then rules with appropriate membership functions.

Fig. 2 shows the normalized input (electrical stimuli) and output (measured neuromuscular response) data.

Fig. 3 shows the two-input ANFIS architecture employed for modeling the knee-joint dynamics.

The structure of ANFIS model implemented is based on:

- First-order Sugeno fuzzy model so the consequent part of fuzzy if-then rules is a linear equation.
- T-norms operator that performs fuzzy AND is algebraic product.
- The type of membership functions (MF) of the inputs are generalized bell functions.

The results of the training process are shown in Figs. 4–6. It is seen that the predicted model correlates very well with the target model. This model will be used throughout for controller design and evaluation. Note that this modeling and control design approach may be applied to any plant for which a mathematical model is not available.

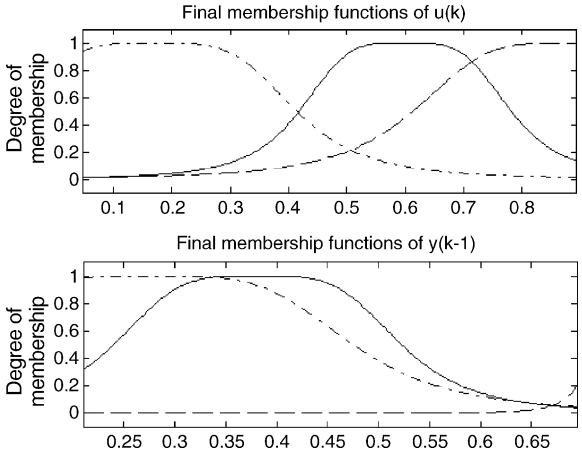

Fig. 4. Initial membership functions for $u(k)$.

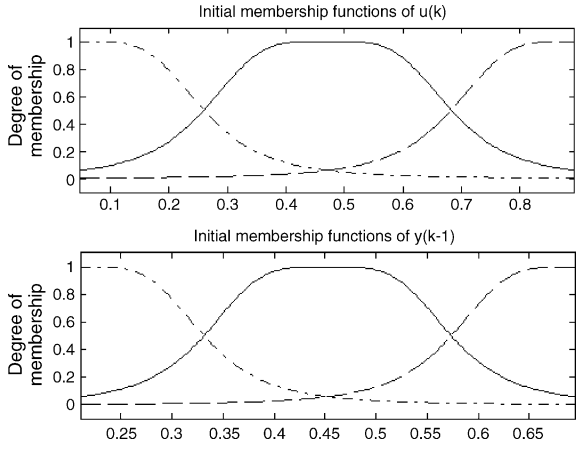

Fig. 5. Initial membership functions for $y(k-1)$.

3. Neural networks based control

Neural networks (NN) represent an important paradigm for classifying patterns or approximating complex non-linear process dynamics. These properties clearly indicate that NN exhibit some intelligent behavior, and are good candidate

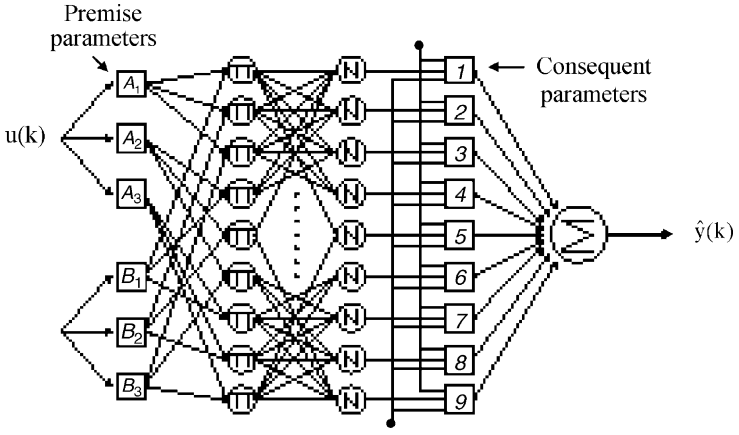

Fig. 3. Structure of the ANFIS model.

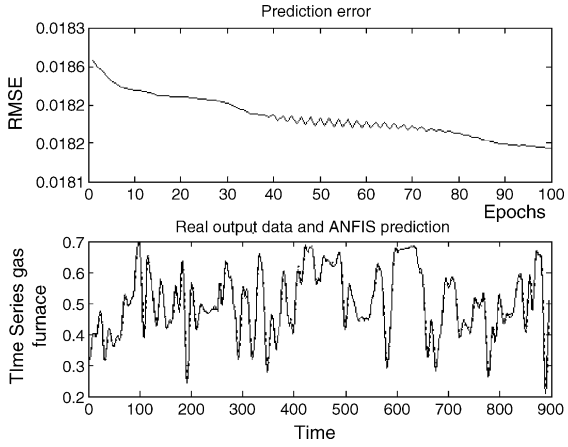

Fig. 6. Comparison of the observed output and ANFIS prediction.

models for non-linear processes, for which no perfect mathematical model is available.

Neural networks have been applied very successfully in the identification and control of dynamic systems. The universal approximation capabilities of the Multi-Layer Perceptron (MLP) have made it a popular choice for modeling non-linear systems and for implementing general-purpose non-linear controllers [5,6].

Prior to control implementation, neural networks have to be trained. This means that, given a set of input–output patterns (called the training set), the connection weights of the neural network are adjusted in order to approximate the input–output patterns provided in the training set according to some predefined criterion. After training, the neural network can be used to predict a new output pattern, based on the input pattern only. The adaptation law that allows adjusting the connection weights is called the learning algorithm.

3.1. Neural network model identification

The identification of a neural network model requires a priori the selection of the inputs to the network and a suitable internal network topology. One of the simplest forms of such topology is an MLP made of three layers:

- input layer (where sets of data are presented to the network);
- hidden layer;
- output layer (that represents the network output).

All neurons from one layer are connected to all neurons in the next layer. Once a network has been structured for a particular application, that network is ready to be trained. To start this process the initial weights are chosen randomly. Then, the training or learning begins. There are two basic approaches to training:

- *Supervised training*: Training consists of presenting input and output data to the network. The network then processes the inputs and compares its resulting outputs against the desired outputs. Errors are then back propagated through the system, causing the system to adjust the weights which control the network. This training is considered complete when the neural network reaches a user defined performance level. When no further training is necessary, the weights are frozen for the underlying application.
- *Unsupervised training*: In unsupervised training, the network is provided with inputs but not with desired outputs. The system itself must then decide what features it will use to group the input data. This is often referred to as self-organization or adaptation.

3.1.1. Neural network forward model

The NN based model structures used for the identification of the neuromuscular system are the non-linear ARX and Output Error models [7]. The predictor is given by

$$\hat{y}(k|\theta) = N[(k,\theta),\theta] \tag{1}$$

where $\varphi(k,\theta)$ is the regression vector which is to be input to the network $N(\cdot)$. The best architecture which gives optimal prediction is the one with six units in the hidden layer. Fig. 7 illustrates the training procedure of the forward neural network model with six hyperbolic tangent neurons in the hidden layer and a linear output neuron.

This NN is trained with the previous input–output data set based on the Levenberg-Marquardt algorithm and the results are shown in Fig. 8. A validation test has been performed in Fig. 9. However, the predictions in this case are relatively poor. Nevertheless, for control purposes the accuracy achieved by the NN forward model is sufficient.

3.1.2. Neural network inverse model

There are two techniques for establishing the inverse model:

- An off-line method known as *general training*.
- An on-line method termed *specialized training*.

In general training (Fig. 10), the objective is to minimize the following performance index:

$$J(\theta) = \frac{1}{2N}\sum_{k=1}^{N}[u(k) - \hat{u}(k|\theta)]^2 \tag{2}$$

An NN architecture with six hyperbolic tangent neurons in the hidden layer and a linear output neuron has been adopted. The NN prediction is shown in Fig. 11.

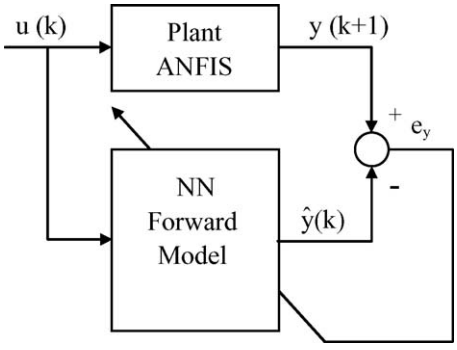

Fig. 7. Training of NN forward model.

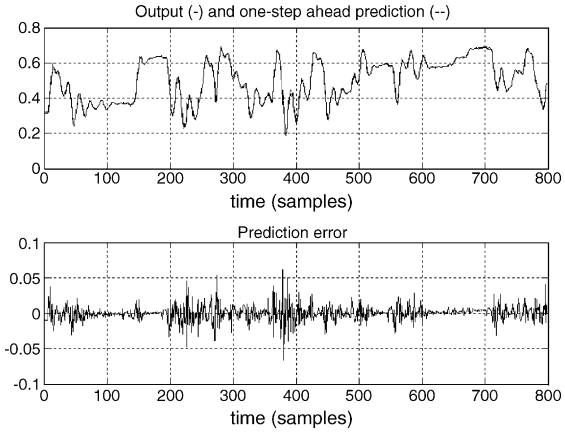

Fig. 8. Comparison of NN one-step ahead prediction with actual output data.

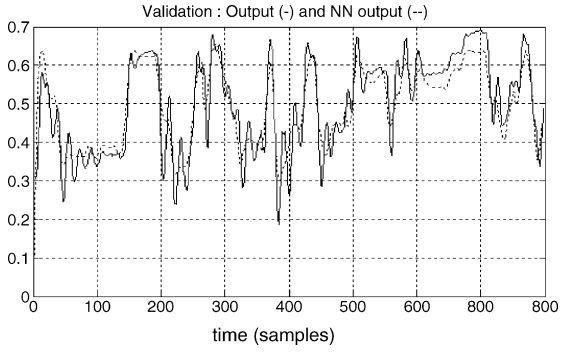

Fig. 9. Validation test on the NN forward model.

3.2. Direct inverse control

When the NN obtained by general training is applied as a controller this resulted in a large deviation between the reference and the output of the system. This is expected since in the general training procedure the minimization of the output error is not taken into account to ensure reference tracking. Steady-state errors can be removed either by including an integral action or an on-line adaptation of the inverse model weights as illustrated by Fig. 12. This leads to an adaptive direct inverse control scheme. Simulation results demonstrated that the adaptive direct inverse controller is able achieve good performance even under inaccurate inverse model as shown in Figs. 13 and 14 where initial NN weights have been set to random values.

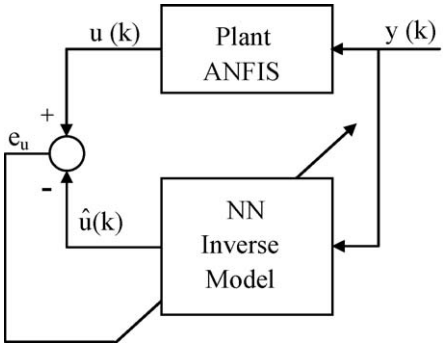

Fig. 10. General training of NN inverse model.

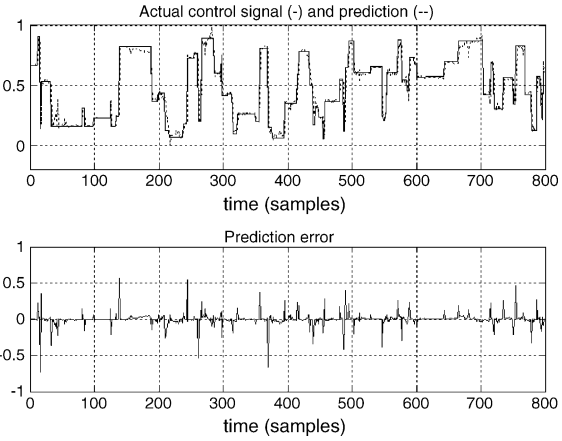

Fig. 11. Comparison of NN prediction with actual input data.

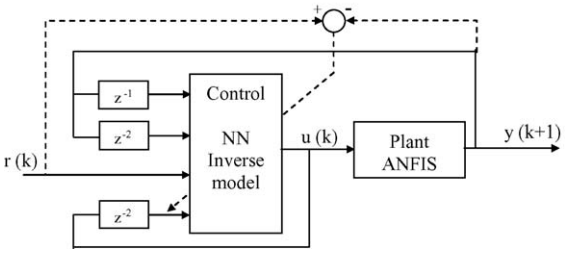

Fig. 12. Adaptive direct inverse control.

Unlike general training, specialized raining is goal directed since it minimizes directly the error measure between the reference and the system output.

$$J(\theta) = \frac{1}{2N}\sum_{k=1}^{N}[r(k) - y(k)]^2 \tag{3}$$

To update NN weights either the steepest descent or back-propagation algorithms can be used. The NN forward model obtained earlier is used to compute the Jacobian matrix based on the approximation $\partial y(k+1)/\partial u(k) \approx \partial \hat{y}(k+1)/\partial u(k)$. Another alternative is to approximate the Jacobian on-line from the changes of plant's inputs and outputs (i.e. $\partial y(k+1)/\partial u(k) \approx \Delta y(k+1)/\Delta u(k)$).

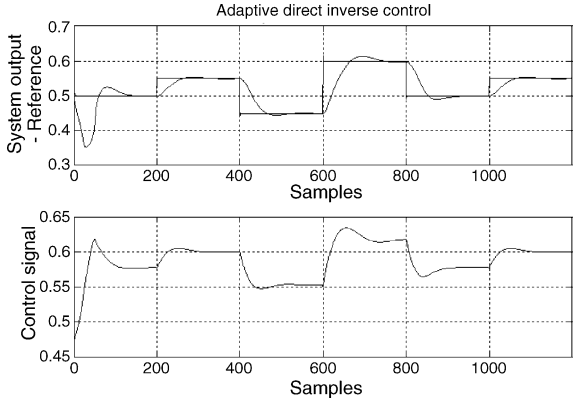

Fig. 13. System response under step changes.

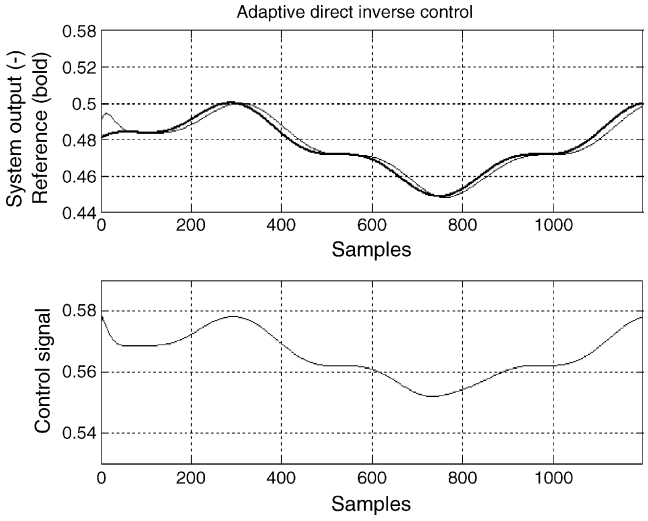

Fig. 14. System response under a two-frequency sinusoidal reference.

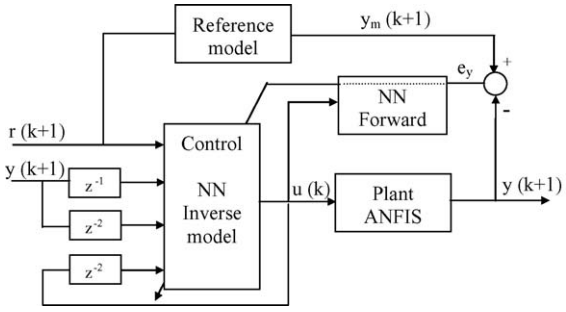

Fig. 15. Direct inverse control based on specialized learning.

The overall control structure is shown in Fig. 15. The reference model is included to provide a desired profile for the reference and to smooth the control signal resulting from the well-known dead beat response produced by the inverse model controller [8].

Fig. 16 demonstrates a perfect model following once convergence of NN inverse model weights has been achieved.

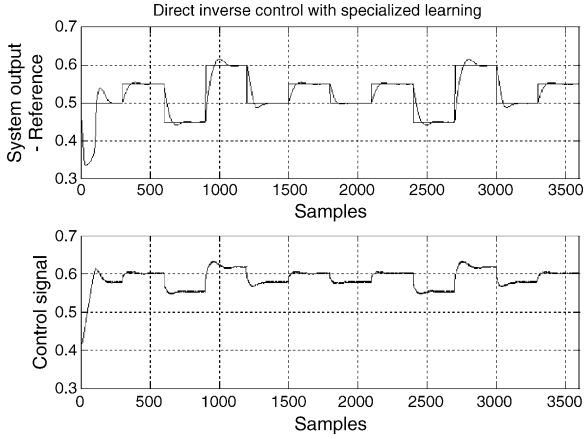

Fig. 16. Direct inverse control based on specialized learning.

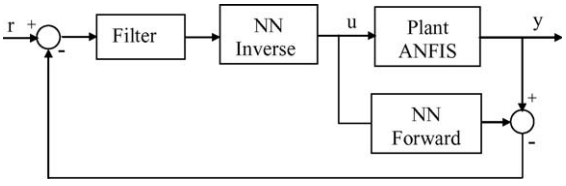

Fig. 17. IMC based on NN models.

3.3. Internal model control

In a classical internal model control (IMC) scheme [9], a plant model is placed in parallel with the actual plant. The difference is used to adjust the command signal. An attractive feature of IMC is that it produces an offset-free response even when the system is subjected to a constant disturbance.

IMC based on NN [10], consists of a NN controller, a NN plant model, and a robustness filter with a single tuning parameter (Fig. 17). The NN controller is generally trained to represent the inverse of the plant. The error between the output of the NN plant model and plant output is used as the feedback input to the robustness filter, which then feeds into the NN controller.

The NN forward and inverse models are those trained previously. The system response under IMC control is shown in Fig. 18 where a disturbance has been applied at time 400. The results demonstrate perfect reference tracking and disturbance rejection.

In Fig. 19, a sinusoidal reference has been applied under the same previous conditions.

The performance of some control algorithms based on neural networks has been evaluated for ANFIS model of the neuromuscular system. An inverse and/or forward model is required prior to the implementation of these neural controllers hence inaccuracies may results in the application stage. Usually, on-line gradient descent weights adaptation improves the response steady-state characteristics. However if the plant exhibits variable dynamics the neural controller performance may deteriorate.

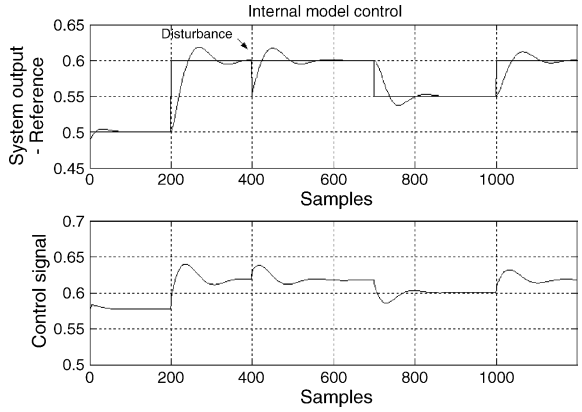

Fig. 18. System response under variable step changes.

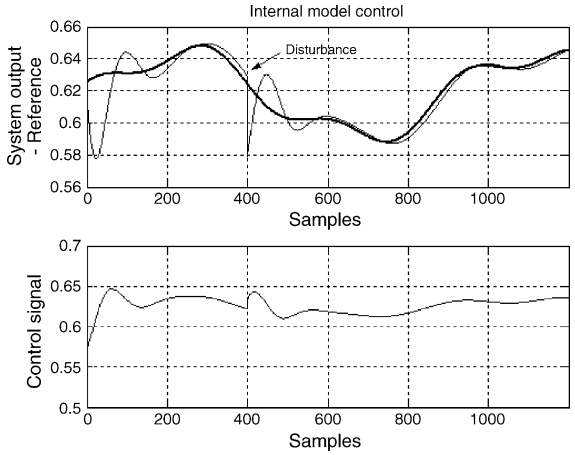

Fig. 19. System response under sinusoidal reference.

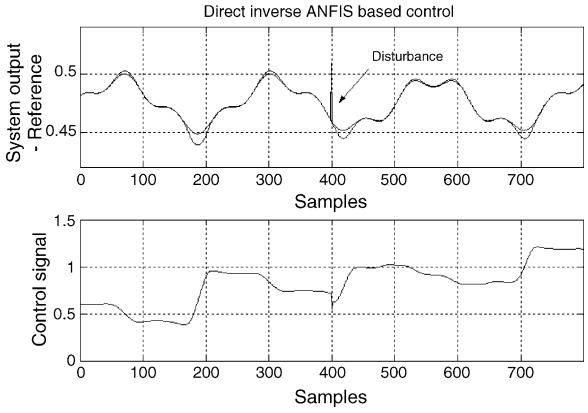

Fig. 21. System response under sinusoidal reference.

4. ANFIS based control

Due to the adaptive capability of ANFIS, its application to adaptive and learning control is immediate. The most common design techniques for ANFIS controllers are derived directly from neural networks counterpart's methodologies. However, certain design techniques are exclusively dedicated to ANFIS [11].

4.1. Direct inverse control

The simplest approach for controller design is a completely open-loop control strategy, in which the controller is the inverse of the process. This method seems straightforward and only one learning task is needed to find the inverse model of the plant. It assumes the existence of the inverse plant, which is not valid in general. Also minimization of the network error does not guarantee minimization of the overall system error as discussed previously.

Inverse learning or general learning for control purpose is performed in two steps:

- In the learning phase, the plant ANFIS inverse model is obtained based on input–output data generated from the former ANFIS model of the system as illustrated by Fig. 20a.
- In the application phase, the obtained ANFIS inverse model is used to generate the control action (Fig. 20b).

As for NN, the trained ANFIS model will have some inaccuracies which may lead to deviations between the reference and the system output. However, it has been noticed that these are relatively smaller in the case of ANFIS inverse control as compared to NN.

The results are illustrated in Fig. 21. Another advantage of ANFIS inverse control over the NN counterpart is that the controller recovers promptly after a disturbance as demonstrated by Fig. 21.

4.2. Inverse control with on-line adaptive learning

When the plant exhibits time variable dynamics, direct inverse control does not guaranty satisfactory response characteristics and steady-state errors. Many of these problems can be overcome using on-line inverse learning strategy. The overall control structure is illustrated in Fig. 22.

On-line learning of ANFIS inverse model occurs at each time step to fine-tune the membership function parameters of ANFIS controller. This control strategy could be applied directly by assuming random premise and consequent parameters or allow for initial learning in open-loop mode and then close the loop once the parameter estimates have assumed some reasonable values.

In either situations control can be initiated with a simple proportional and integral (PI) controller with conservative parameters and then switched to ANFIS controller.

In Figs. 23 and 24 are shown the system responses where the ANFIS controller is initiated with random parameters. The results demonstrate a fast learning leading to satisfactory

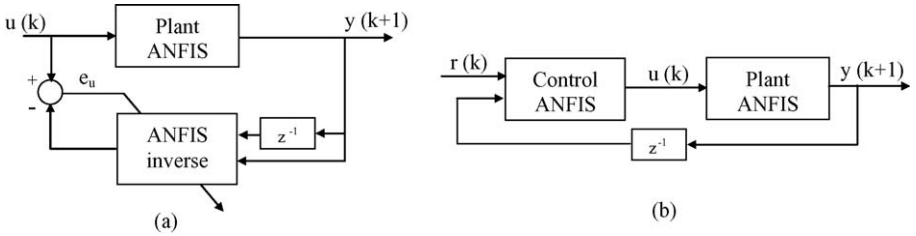

Fig. 20. Block diagram for inverse control method: (a) training phase and (b) application phase [11].

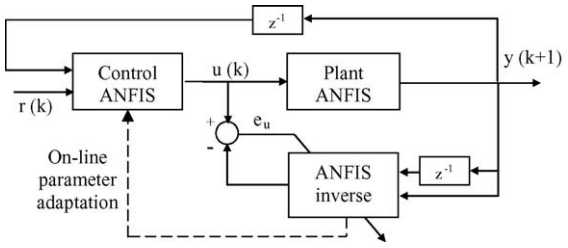

Fig. 22. On-line adaptive learning.

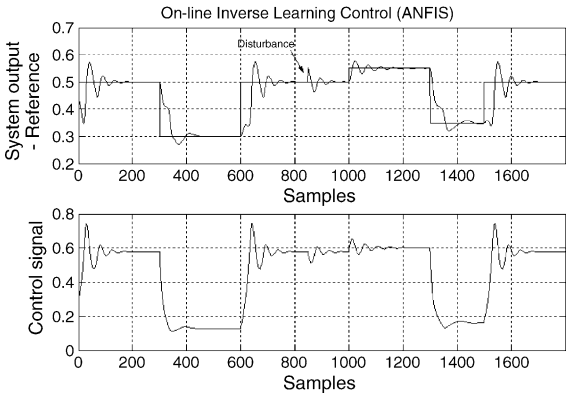

Fig. 23. System response with random initial ANFIS parameters.

transient response characteristics under variable step changes and sinusoidal reference signal, respectively.

Alternatively, initial open-loop identification could be performed prior to closing the loop around the ANFIS controller. This is illustrated in Fig. 25. It can be seen that ANFIS controller achieves satisfactory performance under these conditions.

4.3. Specialized learning control

Specialized learning is an alternative method that tries to minimize the system output error e_y directly by backpropagating error signals through the plant. The overall control system is illustrated by Fig. 26.

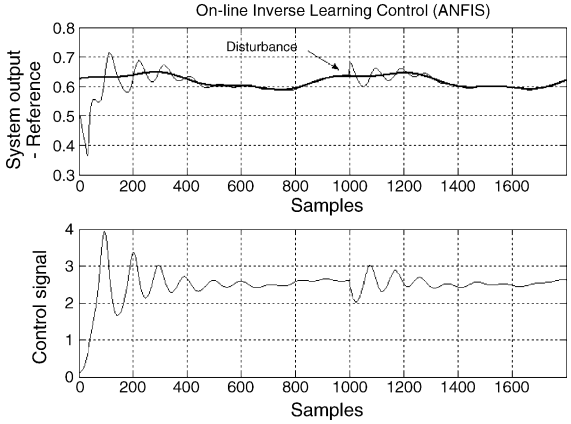

Fig. 24. System response with random initial ANFIS parameters.

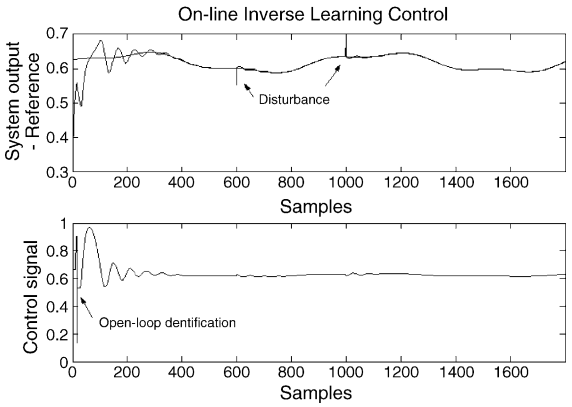

Fig. 25. Initial open-loop identification.

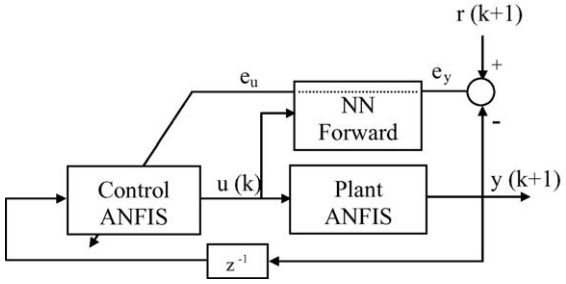

Fig. 26. ANFIS based specialized learning.

Here again, no initial estimates for ANFIS parameters are given. It has been noticed that the first pass in ANFIS hybrid learning rule is decisive. At this stage of the learning process, ANFIS output has converged reasonably well to the target. This explains the fast learning of ANFIS controller (Fig. 27).

Fig. 28 shows the system response to a sinusoidal command signal. The fast recovery following the disturbance is demonstrated.

The results demonstrate the effectiveness of ANFIS-based control strategies. The on-line inverse learning control structure of Fig. 26 does not require an intermediate model

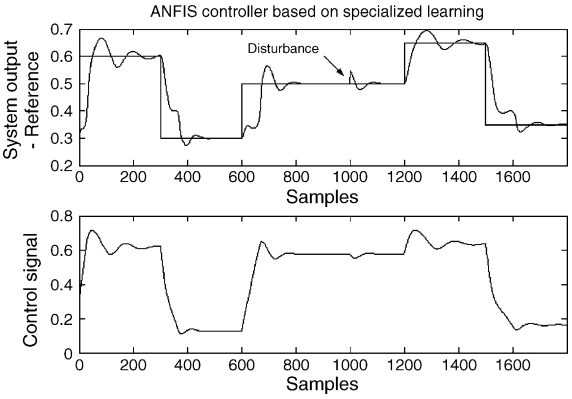

Fig. 27. System response under variable step changes.

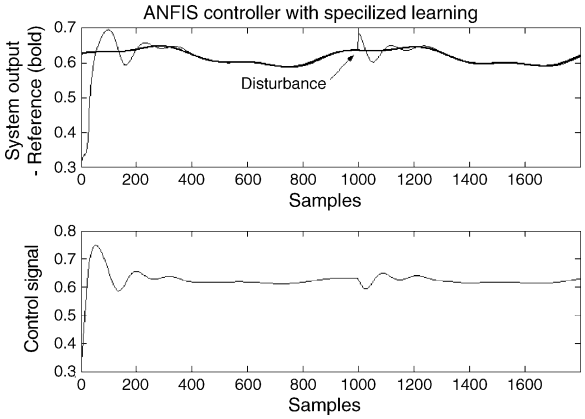

Fig. 28. System response under variable step changes.

of the plant and hence is applicable to any system for which the inverse model may be identified on-line.

5. Overview of the controllers performance and robustness

The control methods presented in this paper are grouped in two parts. The first part deals with NN based controllers while the second part addresses some typical ANFIS based control structures.

NN control has focused on direct inverse control, adaptive direct inverse control, direct inverse with specialized learning control and IMC. Since these control strategies all require a well-trained inverse and/or forward, inaccuracies may result in the implementation stage unless an integral action is incorporated. IMC has a great potential in dealing with uncertainties and is therefore the first line controller within this group.

ANFIS based control methods include direct inverse control, inverse control with on-line adaptive learning, specialized learning control. The on-line adaptive learning does not require any a priori information on the process and therefore is considered as the best controller in this group.

In summary, a general purpose learning control scheme based on ANFIS applicable to systems with unknown dynamics and which are defined only by a set of input–output data is depicted in Fig. 29.

This ANFIS-based control structure does not require any a priori information regarding the plant to be controlled. Identification is performed on the basis of pattern learning by presenting to ANFIS inverse data pairs at each time step and ANFIS control is subsequently updated on-line.

6. Conclusions

Intelligent control is a promising field in modern control technology typically dedicated to highly complex and uncertain systems.

FES is one type of such complex systems and hence may be considered as a challenging system for evaluating the soft computing based control techniques proposed in this work. The aim of the controllers is to adjust the stimulating electrical current to control the knee-joint swing angle.

Neural network-based control has been successfully applied and the potential ability of neural networks to approximate arbitrary non-linearities has been demonstrated.

Although some non-linear control problems can be handled by using these neural control schemes, in situations where there is precise tracking of fast trajectories for non-linear systems with high non-linearities and large uncertainties, the existing neural control schemes are inadequate. The reasons are that the neural controllers based on the gradient method cannot guarantee that the output tracking error converges to zero, and when the controlled systems have large uncertainties, good robustness cannot be obtained either. In other words neural controllers have not a built-in capability to handle system changes.

Adaptive Neuro-Fuzzy Inference Systems are realized by an appropriate combination of neural and fuzzy systems and provide a valuable modeling approach of complex systems. A variety of ANFIS-based control schemes have been reviewed and evaluated. The on-line inverse learning control structure does not require an intermediate model of the plant and hence is

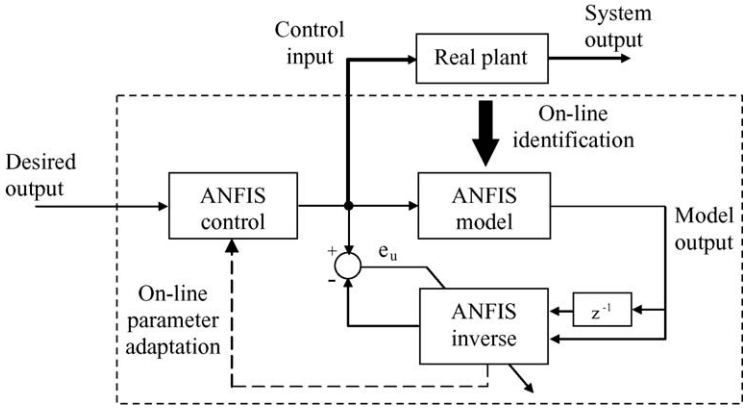

Fig. 29. A general purpose real-time control platform.

applicable to any system for which the inverse model may be identified on-line.

Acknowledgment

The author is grateful to the Deutscher Akademischer Austauschdienst (DAAD) for the financial support of this research work.

Appendix A

A.1. ANFIS architecture

ANFIS architecture consists of five layers with the output of the nodes in each respective layer is represented by $O_{i,l}$ where i is the ith node of layer l [12].

Layer 1: Generate the membership grades

$$\begin{aligned} &O_{1,i} = \mu_{A_i}(x), \quad i = 1,2, \text{ or} \\ &O_{1,i} = \mu_{B_{i-2}}(y), \quad i = 3,4 \end{aligned} \tag{A.1}$$

where x (or y) is the input to the node and A_i (or B_{i-2}) is the fuzzy set associated with this node such us the generalized bell function

$$\mu_{A_i}(x) = \frac{1}{1 + [((x - c_i)/a_i)^2]^{b_i}} \tag{A.2}$$

where $\{a_i, b_i, c_i\}$ is the parameter set referred to as *premise parameters*.

Layer 2: Generate the firing strengths by multiplying the incoming signals and outputs the t-norm operator result, e.g.

$$O_{2,i} = w_i = \mu_{A_i}(x) \times \mu_{B_i}(y), \quad i = 1,2 \tag{A.3}$$

Layer 3: Normalize the firing strengths

$$O_{3,i} = \bar{w}_i = \frac{w_i}{w_1 + w_2}, \quad i = 1,2 \tag{A.4}$$

Layer 4: Calculate rule outputs based on the consequent parameters $\{p_i, q_i, r_i\}$

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i(p_i x + q_i y + r_i) \tag{A.5}$$

Layer 5: Computes the overall output as the summation of incoming signals

$$O_{5,1} = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f}{\sum_i w_i} \tag{A.6}$$

A.2. Hybrid Learning algorithm

- Forward pass

  In the forward pass of the hybrid learning algorithm, node outputs go forward until layer 4 and the consequent are identified by the least-squares method.

  When the values of the premise parameters are fixed, the overall output can be expressed as a linear combination of the consequent parameters

$$\begin{aligned} f &= \frac{w_1}{w_1 + w_2} f_1 + \frac{w_2}{w_1 + w_2} f_2 \\ &= \bar{w}_1 f_1 + \bar{w}_2 f_2 = (\bar{w}_1 x) p_1 + (\bar{w}_1 y) q_1 + (\bar{w}_1) r_1 \\ &\quad + (\bar{w}_2 x) p_2 + (\bar{w}_1 y) q_2 + (\bar{w}_1) r_2 \end{aligned} \tag{A.7}$$

which is linear in the consequent parameters p_1, q_1, r_1, p_2, q_2 and r_2

$$f = [\,\bar{w}_1 x \quad \bar{w}_1 y \quad \bar{w}_1 \quad \bar{w}_2 x \quad \bar{w}_2 y \quad \bar{w}_{21}\,] \begin{bmatrix} p_1 \\ q_1 \\ r_1 \\ p_2 \\ q_2 \\ r_2 \end{bmatrix} = XW \tag{A.8}$$

If X matrix is invertible then

$$W = X^{-1} f \tag{A.9}$$

otherwise a pseudo-inverse is used to solve for W.

$$W = (X^{\mathrm{T}} X)^{-1} X^{\mathrm{T}} f \tag{A.10}$$

- Backward pass

  In the backward pass, the error signals propagate backward and the premise parameters are updated by gradient descent.

$$a_{ij}(t+1) = a_{ij}(t) - \frac{\eta}{p} \frac{\partial E}{\partial a_{ij}} \tag{A.11}$$

where η is the learning rate for a_{ij}. The chain rule is used to calculate the partial derivatives used to update the membership function parameters.

$$\frac{\partial E}{\partial a_{ij}} = \frac{\partial E}{\partial f} \frac{\partial f}{\partial f_i} \frac{\partial f_i}{\partial w_i} \frac{\partial w_i}{\partial \mu_{ij}} \frac{\partial \mu_{ij}}{\partial a_{ij}} \tag{A.12}$$

The partial derivatives are derived as follows:

$$E = \frac{1}{2}(f - f^{\mathrm{t}})^2 \quad \text{hence} \quad \frac{\partial E}{\partial f} = (f - f^{\mathrm{t}}) = e \tag{A.13}$$

$$f = \sum_{i=1}^{n} f_i \quad \text{hence} \quad \frac{\partial f}{\partial f_i} = 1 \tag{A.14}$$

$$\begin{aligned} &f_i = \frac{w_i}{\sum_{i=1}^{n} w_i}(p_i x + q_i y + r_i) \\ &\text{hence} \quad \frac{\partial f_i}{\partial w_i} = \frac{(p_i x + q_i y + r_i) - f}{\sum_{i=1}^{n} w_i} \end{aligned} \tag{A.15}$$

$$w_i = \prod_{j=1}^{m} \mu_{A_{ji}} \quad \text{hence} \quad \frac{\partial w_i}{\partial \mu_{ij}} = \frac{w_i}{\mu_{ij}} \tag{A.16}$$

The last partial derivative depends on the type of membership functions used. The parameters of the other membership functions are updated in the same fashion.

The gradient is then obtained as

$$\frac{\partial E}{\partial a_{ij}} = e\frac{(p_i x + q_i y + r_i) - f}{\sum_{i=1}^{n} w_i}\frac{w_i}{\mu_{A_{ij}}}\frac{\partial \mu_{A_{ij}}}{\partial a_{ij}} \quad \text{(A.17)}$$

$$\frac{\partial E}{\partial b_{ij}} = e\frac{(p_i x + q_i y + r_i) - f}{\sum_{i=1}^{n} w_i}\frac{w_i}{\mu_{B_{ij}}}\frac{\partial \mu_{B_{ij}}}{\partial b_{ij}} \quad \text{(A.18)}$$

References

[1] L.H. Tsoukalas, R.E. Uhrig, Fuzzy Neural Approaches in Engineering, John Wiley & Sons Inc., 1997.
[2] J.S.R. Jang, C.T. Sun, E. Mizutani, Neuro-Fuzzy and Soft Computing, Prentice Hall, New Jersey, 1997.
[3] J.S.R. Jang, ANFIS architecture, in: J.S.R. Jang, C.T. Sun, E. Mizutani (Eds.), Neuro-Fuzzy and Soft Computing, John Wiley & Sons Inc., 1997.
[4] J.S.R. Jang, ANFIS: adaptive-network-based fuzzy inference system, IEEE Trans. Syst. Man Cybern. 23 (1993) 665–685.
[5] K.S. Narendra, K. Parthsarathy, Identification and control of dynamical systems using neural networks, IEEE Trans. Neural Networks 1 (1990) 4–27.
[6] K.J. Hunt, H.D. Sbarbaro, R. Zbikowski, P.J. Gawthrop, Neural networks for control systems—a survey, Automatica 28 (1992) 1083–1112.
[7] M. Norgaard, Neural network based system identification toolbox, Tech. Report 97-E-851, Department of Automation, DTU, Denmark, 1997.
[8] M. Norgaard, Neural network based control system toolkit, Tech. Report 96-E-830, Department of Automation, DTU, Denmark, 1996.
[9] M. Morari, E. Zafiriou, Robust Process Control, Prentice Hall, Englewood Cliffs, 1989.
[10] K.J. Hunt, H.D. Sbarbaro, Neural networks for nonlinear internal model control, IEE Proc. D 138 (1991) 431–438.
[11] J.S.R. Jang, Neuro-fuzzy control I, in: J.S.R. Jang, C.T. Sun, E. Mizutani (Eds.), Neuro-Fuzzy and Soft Computing, John Wiley & Sons Inc., 1997
[12] J.W. Hines, MatLab Supplement to Fuzzy and Neural Approaches in Engineering, John Wiley & Sons Inc., 1997.