Australian Journal of Basic and Applied Sciences, 7(4): 626-640, 2013
ISSN 1991-8178

A PID-like ANFIS Controller Trained by PSO Technique to Control Nonlinear MIMO Systems

[1]Omar F. Lutfy and [2]Enas Tariq Khduir

[1]Control and Systems Engineering Department,
[2]Computer Sciences Department, University of Technology, Baghdad, Iraq.

Abstract: This paper presents a PID-like adaptive neuro-fuzzy inference system (ANFIS) controller that can be trained by the particle swarm optimization (PSO) technique to control nonlinear multi-input multi-output (MIMO) systems. Instead of the hybrid learning methods that are widely used in the literature, the PSO is used to adjust all of the ANFIS parameters along with the input and output scaling factors for this controller. Two nonlinear MIMO plants have been selected to be controlled by this controller. The simulation results show the notable control accuracy and generalization ability of this MIMO ANFIS controller. In addition, the controller robustness to output disturbances has been also tested and the results clearly indicate the effectiveness of this controller. Finally, a comparative study with a real-coded genetic algorithm (GA) and a recently developed optimization technique, namely the global-best harmony search (GHS), shows the superiority of the PSO over these two optimization methods in terms of control accuracy and processing time.

Key words: Adaptive neuro-fuzzy inference system, particle swarm optimization, genetic algorithm, global-best harmony search, multi-input multi-output systems.

INTRODUCTION

In recent years it has been recognized that to realize more flexible control systems it is necessary to incorporate other elements, such as logic, reasoning and heuristics into the more algorithmic techniques provided by conventional control theory, and such systems have come to be known as intelligent control systems. Intelligent control is the discipline in which control algorithms are developed by emulating certain characteristics of intelligent biological systems. It is quickly emerging as a technology that may open avenues for significant advances in many areas (Linkens and Nyongesa, 1996). One of the most widely used intelligent systems is the ANFIS network which was proposed by Jang (Jang, 1993; Jang *et al.*, 1997). To solve a particular problem, the training method plays an important role in determining the final performance of the ANFIS network. In this regard, several ANFIS training methods have been proposed in the literature. For instance, Jang (1993) proposed a hybrid learning rule which combines the gradient descent technique and the least square estimator (LSE) method. Lin (2004) combined the GA and the LSE in a hybrid method to optimize the ANFIS parameters. Lutfy *et al.* (2009) used the GA to train all of the ANFIS parameters acting as a feedback controller to control nonlinear systems. Despite the fact that both the GA and the PSO methods have several similarities, PSO does not have complicated evolutionary operators such as the crossover and the mutation in the GA. Therefore, the computational complexity is less in the PSO technique than the GA, and this will make the PSO more suitable for the ANFIS training task, as will be seen in the comparative study presented in the results and discussion section of this work. Ghomsheh *et al.* (2007) used the PSO with some modifications, which are inspired by natural evolutions, to train the ANFIS structure. They used this ANFIS as an identifier to identify nonlinear dynamical systems. Zhao *et al.* (2009) proposed a two-stage method to extract Takagi-Sugeno fuzzy models from data using subtractive clustering and coevolutionary particle swarm optimization (CPSO). As in (Ghomsheh *et al.*, 2007), they also used the ANFIS as an identifier. In (Ghomsheh *et al.*, 2007) and (Zhao *et al.*, 2009) the standard PSO technique has been adapted to improve its performance; however, this adaptation required additional computation complexity which degraded the standard PSO computational simplicity. It is well known that the hybrid learning rule is the most commonly used technique to train the ANFIS network. As a supervised learning method, this hybrid learning rule requires a teaching signal in its operation. However, in control system design, when the ANFIS network is to be utilized as a feedback controller, this teaching signal is difficult to be provided, since the desired control actions, which represent the teaching signal in this case, are simply unknown. In order to mitigate this difficulty, several ANFIS learning methods have been proposed in the literature. For instance, Djukanović *et al.* (1997) utilized a special ANFIS learning technique, called temporal back propagation (TBP), to control nonlinear MIMO systems. The basic idea of this method is to consider both the controller and the plant as a single unit at each time step. Utilizing this method, several ANFIS controllers were adopted to handle different operating points and a neural network classifier was used to select the best controller for a particular operating point. However, this ANFIS training method is characterized by its heavy computational load and the complexity in implementation.

Corresponding Author: Omar F. Lutfy, Control and Systems Engineering Department, Computer Sciences Department, University of Technology, Baghdad, Iraq.

Another learning approach to train the ANFIS as a controller is represented by the inverse learning method, where the ANFIS network is trained to learn the inverse dynamics of the controlled plant. This method has been successfully applied to control nonlinear MIMO systems (Toha and Tokhi, 2009; Yao and Chai, 2007; Zhou and Jagannathan, 1996). However, the effectiveness of this ANFIS learning method depends on three essential issues. Firstly, the existence of an accurate model of the original system, which can not be easily developed for complex systems; secondly, the availability of the inverse dynamics of the system, which does not always exist; and thirdly, the appropriate distribution of the training data, more specifically the training data should be uniformly distributed across the input space of the controller. However, such kind of data distribution might not be available due to either the scarcity of the data or the constraints imposed by the system dynamics. Besides the above mentioned methods, another training method to train the ANFIS as a controller for nonlinear MIMO systems was used in (Mahmoud *et al.*, 2010; Cao *et al.*, 2007; Touati *et al.*, 2002). In this method, the ANFIS network is trained by using data collected from another working controller. The drawback of this method is that the efficiency of the resulting ANFIS controller depends heavily on the performance of the original controller, which is mostly a linear one.

In this work, the standard PSO technique, which does not rely on a teaching signal, is used to optimize both the antecedent and the consequent parameters of the ANFIS network to act as a PID-like feedback controller to control nonlinear MIMO systems. In addition, the input and output scaling factors for this controller are also obtained by the PSO technique. In order to verify the effectiveness of the PSO technique, a comparative study is made with other optimization methods namely, a real-coded GA and a GHS technique.

MATERIALS AND METHODS

Structure of the PID-like ANFIS Controller:

To describe the structure of the PID-like ANFIS controller, shown in Fig. 1, let $\bar{x}_1$, $\bar{x}_2$, and $\bar{x}_3$ represent the three input variables e, Δe, and δe, respectively, and y represents the single output variable u of this controller, where e is the error of the plant being controlled, Δe is its rate of change, δe is the summation of errors, and u is the output control action of the ANFIS controller. Five fuzzy linguistic terms are used for each input variable, let the set $\{A_1, A_2, A_3, A_4, A_5\}$ represents these terms, where A_1, A_2, A_3, A_4, and A_5 represent negative big, negative small, zero, positive small, and positive big, respectively. The structure of the ANFIS network is based on Sugeno fuzzy models. A typical Sugeno fuzzy rule can be expressed in the following form:

If $\bar{x}_1$ is A_{a1} and $\bar{x}_2$ is A_{a2} and $\bar{x}_3$ is A_{a3} Then $y=k_0+k_1\bar{x}_1+k_2\bar{x}_2+k_3\bar{x}_3$, where $a1, a2, a3 \in \{1, 2, 3, 4, 5\}$.

In Sugeno fuzzy model, y in the above rule can be either a constant or a linear function of the input variables. When y is a constant, a zero-order Sugeno fuzzy model is obtained in which, the consequent of a rule is specified by a singleton. When y is a first-order polynomial, as shown in the above rule, a first-order Sugeno fuzzy model is obtained. In this work, the zero-order Sugeno fuzzy model is used. This selection was made in order to reduce the number of parameters to be optimized by the PSO in the consequent part of the MIMO ANFIS controller from 1000 parameters in the case of a first-order Sugeno fuzzy model into 250 parameters in the case of the zero-order model. From the simulation results of this work, it can be concluded that the zero-order model has achieved a satisfactory performance, which justify the selection of this model. In order to describe the function of each layer in the ANFIS network, the output of the i^{th} node in layer k will be expressed as $O_{k,i}$.

Layer 1: All nodes in this layer are adaptive nodes, and they generate the degree of membership for each of the three input variables. The node function is:

$$\begin{aligned} O_{1,i} &= \mu_{A_i}(x_1), && i = 1,2,...,5 \\ O_{1,i} &= \mu_{A_{i-5}}(x_2), && i = 6,7,...,10 \\ O_{1,i} &= \mu_{A_{i-10}}(x_3), && i = 11,12,...,15 \end{aligned} \tag{1}$$

The membership functions of A_a, a = 1, 2, …, 5, for each input variable are chosen to be bell-shaped activation functions. Only two parameters have been used for each of these functions instead of the widely used generalized bell function, which uses three parameters, in order to reduce the number of parameters to be optimized by the PSO technique. These bell-shaped functions can be represented by:

$$\mu_{A_a}(x_k) = \exp\left(-1/2\left(\frac{x_k - C_{A_a}^{x_k}}{\sigma_{A_a}^{x_k}}\right)^2\right) \tag{2}$$

where x_k, $k \in \{1,2,3\}$, represents the scaled input variables after they have been multiplied by the input scaling factors (c_1 for the error, c_{11} for its rate of change, and c_{111} for the summation of errors, as shown in Fig. 1). $C_{A_a}^{x_k}$ and $\sigma_{A_a}^{x_k}$ are the centers and the widths of these bell-shaped functions, respectively.

Layer 2: The nodes in this layer perform the multiplication operation for fuzzy inferencing. The number of all possible antecedent combination is 125. Therefore, there are 125 nodes in this layer. Each node output represents the activation level of a rule:

$$O_{2,i} = w_i = \mu_{A_{a1}}(x_1) \cdot \mu_{A_{a2}}(x_2) \cdot \mu_{A_{a3}}(x_3) \tag{3}$$

where i=1, 2, …, 125 and $a1$,$a2$,$a3$, $\in \{1,2,3,4,5\}$.

Layer 3: As in layer 2, this layer has 125 nodes. The i^{th} node calculates the ratio of the i^{th} rule's firing strength to the sum of all rule's firing strengths:

$$O_{3,i} = \overline{w}_i = \frac{w_i}{\sum_{j=1}^{125} w_j} \quad \text{, where } i=1, 2, \ldots, 125. \tag{4}$$

Layer 4: Similar to layers 2 and 3, there are 125 nodes in this layer. All the nodes are adaptive nodes. The output of each node is given by:

$$O_{4,i} = \overline{w}_i \cdot k_{0i} \tag{5}$$

where i=1, 2, …, 125, $\overline{w}_i$ is the ith output from layer 3, and k_{0i} is the ith consequent parameter.

Layer 5: The single node in this layer computes the overall output as the summation of all incoming signals:

$$O_5 = \sum_{i=1}^{125} O_{4,i} \tag{6}$$

And finally, O_5 is multiplied by a factor c_2, which represents the output scaling factor of this controller:

$$y = u = c_2 \cdot O_5 \tag{7}$$

Equation 7 above can be written in more detail as:

$$y = u = c_2 \left[\frac{\sum_{i=1}^{R} k_{0i} \cdot \prod_{j=1}^{n} \exp\left(-1/2\left(\frac{(c_j \overline{x}_j) - C_{A_a}^{x_j}}{\sigma_{A_a}^{x_j}}\right)^2\right)}{\sum_{i=1}^{R} \prod_{j=1}^{n} \exp\left(-1/2\left(\frac{(c_j \overline{x}_j) - C_{A_a}^{x_j}}{\sigma_{A_a}^{x_j}}\right)^2\right)} \right] \tag{8}$$

where R=125, n=3, and c_j, j = 1,2,3, represents the three input scaling factors c_1, c_{11}, and c_{111}, respectively. In this work, two ANFIS controllers are required in order to control MIMO plants, as can be seen from Fig. 2. Each of these two controllers has exactly the same structure described above.

An Overview of the Particle Swarm Optimization:

The PSO technique is one of the modern heuristic algorithms, which was first introduced in (Kennedy and Eberhart, 1995). This technique can generate a high-quality solution within shorter calculation time and stable convergence characteristic than other stochastic methods (Gaing, 2004). Each particle keeps track of its coordinates in the problem space, which are associated with the best solution (fitness function) it has achieved

so far. This value is called *pbest*. Another value is the overall best value, and its location, obtained so far by any particle in the group, and it is called *gbest*. The modified velocity and position of each particle can be calculated using the following formulas (Gaing, 2004; Song *et al.*, 2007):

$$v_{j,g}^{(t+1)} = w \cdot v_{j,g}^{(t)} + c_1 * rand() * (pbest_{j,g} - x_{j,g}^{(t)}) + c_2 * Rand() * (gbest_g - x_{j,g}^{(t)}) \quad (9)$$

$$x_{j,g}^{(t+1)} = x_{j,g}^{(t)} + v_{j,g}^{(t+1)} \quad (10)$$

where j=1,2,…,n, g=1,2,…,m, n is the number of particles in a group, m is the number of members in a particle, t is the pointer of iterations, $v_{j,g}^{(t)}$ is the velocity of particle j at iteration t, $x_{j,g}^{(t)}$ is the current position of particle j at iteration t, w is the inertia weight factor, c_1 and c_2 are the acceleration constants, *rand()* and *Rand()* are random numbers between 0 and 1, $pbest_j$ is the pbest of particle j and *gbest* is the gbest of the group.

Implementation of the PSO Algorithm to Train the MIMO ANFIS Controller:

In this work, MATLAB software has been used for the implementation of the PSO technique in order to train the MIMO ANFIS controller by a specifically written program for this purpose. As mentioned before, two separate ANFIS controllers have been used as feedback controllers to control the MIMO systems, see Fig. 2. Therefore, 318 members are required in each particle of the swarm. In this paper, the term "individual" will be used instead of "particle" and "population" instead of "group". The proposed searching procedure to optimize the ANFIS parameters is summarized in the following:

***Step* 1:** Initialize the PSO parameters: w, c_1, c_2, population size, and the maximum number of iterations.

***Step* 2:** Generate randomly the initial population within certain bounds, in which each individual represents the entire antecedent and consequent parameters along with the input and output scaling factors for the two ANFIS controllers. In addition, initialize the velocities, the *pbests*, and the *gbest* for this population.

***Step* 3:** Evaluate the objective function for each individual in the population using the 0.5ISE, which has the following form:

$$0.5ISE = 0.5\sum_{k=0}^{T} e_1^2(k) + e_2^2(k) \quad (11)$$

where $e_1(k) = r_1(k) - y_1(k)$, $e_2(k) = r_2(k) - y_2(k)$ and T is the observation time.

***Step* 4:** Compare each individual's fitness function with its *pbest*. The best fitness function among the *pbests* is denoted as *gbest*.

***Step* 5:** Modify the member velocity of each individual according to (9), and then modify the member position of each individual according to (10).

***Step* 6:** Make sure that the value of each member in each individual does not exceed the corresponding range assigned for that member by the following formula:

$$\begin{aligned} &If\ x_{j,g}^{(t+1)} > X_g^{\max},\ then\ x_{j,g}^{(t+1)} = X_g^{\max} \\ &If\ x_{j,g}^{(t+1)} < X_g^{\min},\ then\ x_{j,g}^{(t+1)} = X_g^{\min} \end{aligned} \quad (12)$$

where $X_g^{\max}$ and $X_g^{\min}$ represent the upper and the lower bounds, respectively, of member g of the individual x_j.

***Step* 7:** Stop if the maximum number of iterations is reached, otherwise increase the iteration counter by one and go back to step 3.

Implementation of the Real-coded GA to Train the MIMO ANFIS Controller:

Utilizing the MATLAB software, a specific program has been written in an m-file to implement the real-coded GA for training the MIMO ANFIS controller. The real-coded GA used in this work includes four operators, namely; hybrid selection, elitism, crossover, and mutation operators (Lutfy *et al.*, 2009). The hybrid selection operator (Al-Said, 2000) is a combination of roulette wheel and deterministic selection. For a particular generation in this method, only those chromosomes that have better fitness values than the worst individual in the old population are accepted in the new population. In the elitism operator, the best two parents in a given generation are copied directly into the next generation as they are in order to prevent the best fitness value in a given generation from becoming worse than that in the previous one (Sivanandam and Deepa, 2008).

In the real-coded crossover operator, a pair of mating chromosomes exchanges information by exchanging a subset of their components, where an integer position k is selected uniformly at random along the chromosome length. Then two new chromosomes are created by swapping all the genes between positions k+1 and L, where L is the chromosome length (Sivanandam and Deepa, 2008). For example, the pair of chromosomes x and y as: x=[1.2,3,7,5,6.1] and y=[9.5,8,3,1,9] are crossed over at the fourth digit position to yield: x'=[1.2,3,7,1,9] and y'=[9.5,8,3,5,6.1]. Finally, the task of the real-coded mutation operator is to cause random changes in the components of the new population chromosomes by replacing the mutated 'gene' with another random number selected in the same range assigned for that 'gene'. For instance, the chromosome z=[3,1,9,8.5,7] is mutated at the second 'gene' to yield: z'=[3,1.6,9,8.5,7]. The genetic learning procedure used to train the MIMO ANFIS controller is summarized in the following steps:

***Step* 1:** Initialize the crossover probability (P_c), the mutation probability (P_m), the population size, and the maximum number of generations.

***Step* 2:** Generate randomly the initial population within certain bounds, in which each chromosome represents the entire antecedent and consequent parameters along with the input and output scaling factors for the two ANFIS controllers.

***Step* 3:** Evaluate the objective function for each chromosome in the population using the 0.5ISE criterion defined previously in (11).

***Step* 4:** Apply the elitism strategy described before.

***Step* 5:** Select two individuals by using the hybrid selection method, and then apply the real-coded genetic operators of crossover and mutation described previously to form two new chromosomes.

***Step* 6:** Put the resulting two chromosomes in the new population.

***Step* 7:** Repeat Step 5 until all the chromosomes in the new population are created, i.e. until the new population size is equal to the initial (old) population size.

***Step* 8:** Replace the initial (old) population with the new population.

***Step* 9:** Stop if the maximum number of generations is reached, otherwise increase the generation counter by one and go back to step 3.

Implementation of the GHS to Train the MIMO ANFIS Controller:

The harmony search (HS) is a newly developed optimization algorithm. It was first introduced in (Geem *et al.*, 2001) by an attempt to mimic the music improvisation process. Similar to the PSO, the HS algorithm is simple in concept and easy in implementation. Utilizing the MATLAB software, a specific program has been written for the GHS, which is a modified version of the standard HS algorithm, to train the ANFIS as a MIMO feedback controller using the following learning procedure (Coelho and Bernert, 2009; Zarei *et al.*, 2009; Omran and Mahdavi, 2008):

***Step* 1:** Initialize the GHS parameters, which include; the harmony memory size (*HMS*), or the number of solution vectors in the harmony memory (HM), the harmony memory considering rate (*HMCR*), the pitch adjusting rate (*PAR*), and the number of improvisations (*NI*).

***Step* 2:** Initialize the HM, which is a memory location used to store all the solution vectors, by randomly generating *HMS* solution vectors within certain bounds. Each of these solution vectors represents the entire antecedent and consequent parameters along with the input and output scaling factors for the two ANFIS controllers.

***Step* 3:** Evaluate the objective function for each vector in the HM using the 0.5ISE criterion defined previously in (11).

***Step* 4:** Search for the index of best and worst harmonies in the HM, and call them best_index and worst_index, respectively.

***Step* 5:** From the HM, improvise a new harmony vector, $x_g^{'} = (x_1^{'}, x_2^{'}, ..., x_m^{'})$, where m represents the number of decision variables in the problem being optimized. The improvisation is done by applying three rules, namely: (1) memory consideration, (2) pitch adjustment, and (3) random selection. In memory consideration, the value of a given decision variable in the new harmony vector ($x_g^{'}$), where $1 \le g \le m$, is selected from any of the values for that decision variable in the HM range ($x_{1g}, x_{2g}, ..., x_{HMS,g}$). The *HMCR*, which varies between 0 and 1, is the rate of choosing one value from the historical values stored in the HM for a given decision variable. On the other hand, (1-*HMCR*) is the rate of randomly selecting one value from the possible range of values for that decision variable, as shown in the following:

if (*rand* ( ) ≤ *HMCR*)

$x_g^{'} \leftarrow x_g^{'} \in \{x_{1g}, x_{2g}, ..., x_{HMS,g}\}$

else

$x_g^{'} \leftarrow x_g^{'} \in X_g^{'}$

end

where *rand* ( ) is a random number between 0 and 1 and $X_g^{'}$ is a random value from the range assigned for member *g*. After this process, every component obtained by the memory consideration is checked to determine whether it should be pitch-adjusted or not. This operation uses the *PAR* parameter, which is the rate of pitch adjustment as follows:

if (*rand* ( ) ≤ *PAR*)

$x_g^{'} = x_g^{'} \pm rand\ (\ ) \times bw$

else

$x_g^{'} = x_g^{'}$

end

where *bw* is an arbitrary distance bandwidth. In the classical HS algorithm, selecting a suitable value for the *bw* parameter represents a difficult problem. This difficulty poses a major drawback of the classical HS algorithm, especially when the problem being optimized involves many decision variables, as in the case of optimizing the ANFIS parameters considered in this work. In order to alleviate this difficulty, a new modified version of HS algorithm was proposed by Omran and Mahdavi (2008). Inspired by some concept borrowed from PSO algorithm, this new HS version modifies the pitch-adjustment step such that the new harmony can mimic the best harmony in the HM. Therefore, this HS version was called global-best harmony search (GHS). In particular, the *bw* parameter is totally eliminated in the GHS, as shown in the following:

if (*rand* ( ) ≤ *PAR*)

$x_g^{'} = x_{best,g}$

else

$x_g^{'} = x_g^{'}$

end

where (*best*) is the index of the best harmony in the HM.

***Step* 6:** Update the HM. If the newly improvised harmony vector has a better fitness function compared to the worst harmony in the HM, it replaces that worst harmony.

***Step* 7:** Stop if the maximum number of improvisations, *NI,* is reached, otherwise go back to Step 4.

RESULTS AND DISCUSSION

In order to demonstrate the effectiveness of the proposed MIMO ANFIS controller, which acts as a feedback controller as shown in Fig. 2, two nonlinear MIMO plants are selected to be controlled by this controller. The aim is to test the ability of this controller in decoupling of loop-interactions while tracking the given reference inputs $r_1(k)$ and $r_2(k)$. Therefore, the training and testing signals were deliberately chosen as shown in Fig.s 3 (a, b) and 4 (a, b) respectively. From these figures, it is clear that there is a difference between the training and the testing signals. This difference is important in testing the generalization ability of this controller. The PSO parameters are set to the following values: group size 50; maximum number of iterations: 300, c_1: 2.0, c_2: 2.0, and *w*: 0.5. From the simulation tests, these values were found to be sufficient for training the MIMO ANFIS controller presented in this work to control the following nonlinear MIMO plants:

Plant 1:

A multivariable nonlinear plant is described by the following (Narendra and Parthasarathy, 1990; Belikov and Petlenkov, 2008):

$$\begin{bmatrix} y_1(k) \\ y_2(k) \end{bmatrix} = \begin{bmatrix} \dfrac{y_1(k-1)}{1+y_2^2(k-1)} \\ \dfrac{y_1(k-1)y_2(k-1)}{1+y_2^2(k-1)} \end{bmatrix} + \begin{bmatrix} u_1(k-1) \\ u_2(k-1) \end{bmatrix} \tag{13}$$

Fig. 4 shows the output responses, control actions and the best 0.5ISE against the iterations for this plant. Fig.s 4 (a) and 4 (b) show that the proposed MIMO PID-like ANFIS controller has performed well in both tracking and de-coupling in controlling this MIMO plant, where the tracking of the two reference signals has been achieved with zero steady-state error and very small overshoot at the beginning of each step change. Fig.s 4 (c) and 4 (d) show that there is a sharp control action when each reference input is changed. This action is necessary to compensate for the interactions between the loops. Bearing in mind the difference between the training and the testing signals, it can be concluded that this controller has achieved good generalization ability. From Fig. 4 (e), it can be seen that the 0.5ISE has reached its near optimal value in less than the early 40 iterations. Therefore, the choice of the maximum number of iterations to be 300 seems to be adequate. This fact indicates the fast convergence to the optimal solution that has been achieved by the PSO in training this controller. In order to compare the results of controlling this MIMO plant with other works, the same reference signals used in (Belikov and Petlenkov, 2008) are also adopted here with the same settings for the PSO parameters described above, and the results are shown in Fig. 5. Compared to the results presented in (Belikov and Petlenkov, 2008), Fig. 5 (a) shows that the MIMO PID-like ANFIS controller has achieved better tracking accuracy to the reference signals. It is worth to highlight that the control design method used in (Belikov and Petlenkov, 2008) was based on the analytical calculation of control signals using a neural network as an identifier to the controlled system. As a consequence, the effectiveness of this control design approach depends on the identifier accuracy in representing the identified system. On the other hand, the MIMO PID-like ANFIS controller presented in this work is characterized by its simplicity in implementation with no need for an identifier to the system to be controlled. Fig. 5 (b) shows the control signals produced by the MIMO PID-like ANFIS controller to effectively handle each step change in the two reference signals.

Plant 2:

This nonlinear MIMO plant is described by the following (Song and Li, 2006; Petlenkov, 2007):

$$\begin{aligned} y_1(k) &= \frac{0.7y_1(k-1)y_1(k-2)}{1+y_1^2(k-1)+y_2^2(k-2)} + 0.3u_1(k-2) + u_1(k-1) + 0.2u_2(k-2) \\ y_2(k) &= \frac{0.5y_2(k-1)\sin(y_2(k-2))}{1+y_2^2(k-1)+y_1^2(k-2)} + 0.5u_2(k-2) + u_2(k-1) + 0.2u_1(k-2) \end{aligned} \tag{14}$$

Fig. 6 shows the output responses, control actions and the best 0.5ISE against the iterations for this plant. Despite the complexity of this plant, the MIMO PID-like ANFIS controller has achieved good de-coupling and tracking to the desired responses (see Fig.s 6 (a) and 6 (b)) with zero steady-state error and some oscillations at the start of each change in the testing signals. Although the training signals are different from the testing signals, the MIMO ANFIS controller has successfully generalized its learning to effectively deal with unexpected testing signals. This performance for the controller is due to its control actions given in Fig.s 6 (c) and 6 (d). As in the first plant, the 0.5ISE has reached to its near optimal value in less than the early 40 iterations, as shown in Fig. 6 (e).

Comments on Handling the Coupling Effects:

In the above mentioned MIMO plants, the loop-interactions can be observed from the dependency of one output on the previous state of the other output, as indicated in equation (13), and moreover, on the previous state of the other input, as indicated in equation (14). For each of these two plants, the training (and testing) signals for each of the two loops were selected to be the inverse of each other. This selection was made in order to evaluate the controller ability in handling difficult type of inputs that stimulate the coupling effects inherited in the MIMO plant. In this work, the controller efficiency to cope with these coupling problems is assessed by the resulting system response to the testing signals. It is worth mentioning that in some reported works (Mahmoud *et al.*, 2010; Lasheen *et al.*, 2009; Mahmoud *et al.*, 2008; Chopra *et al.* 2007), two separate decoupling controllers, beside the two main controllers, were utilized to handle the loop-interactions problem. However, in this work the reliance was only on the two ANFIS controllers to handle this coupling issue.

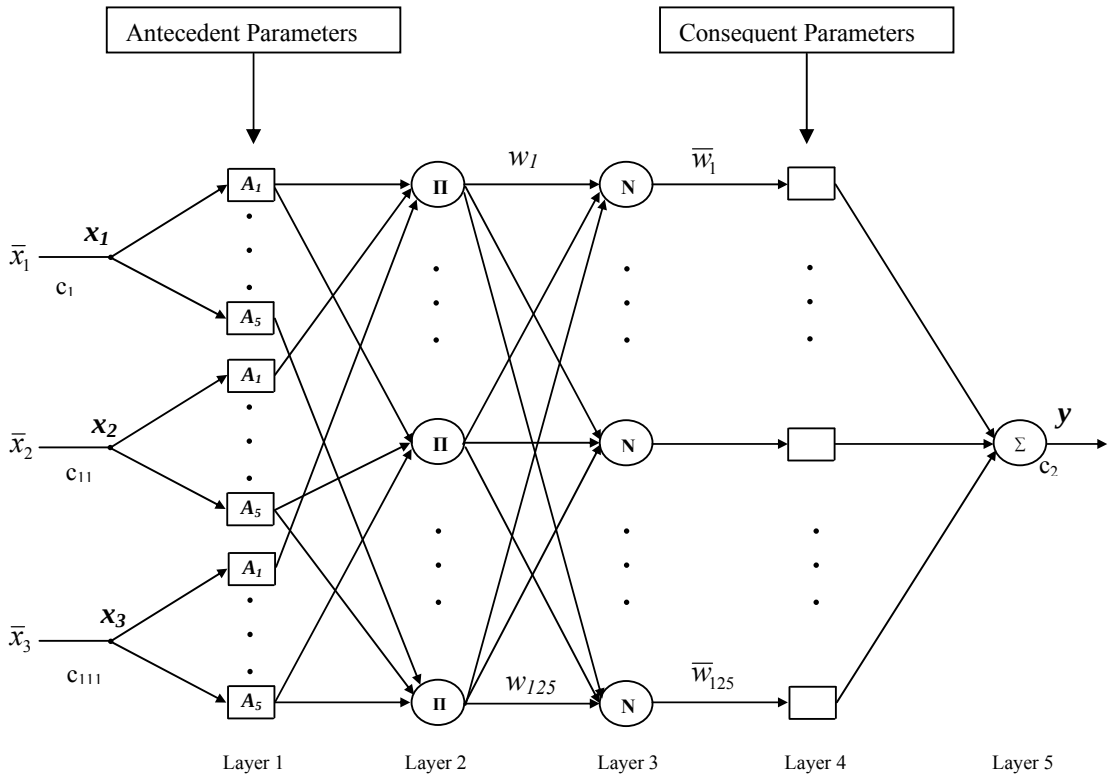


Fig. 1: Structure of the PID-like ANFIS controller.

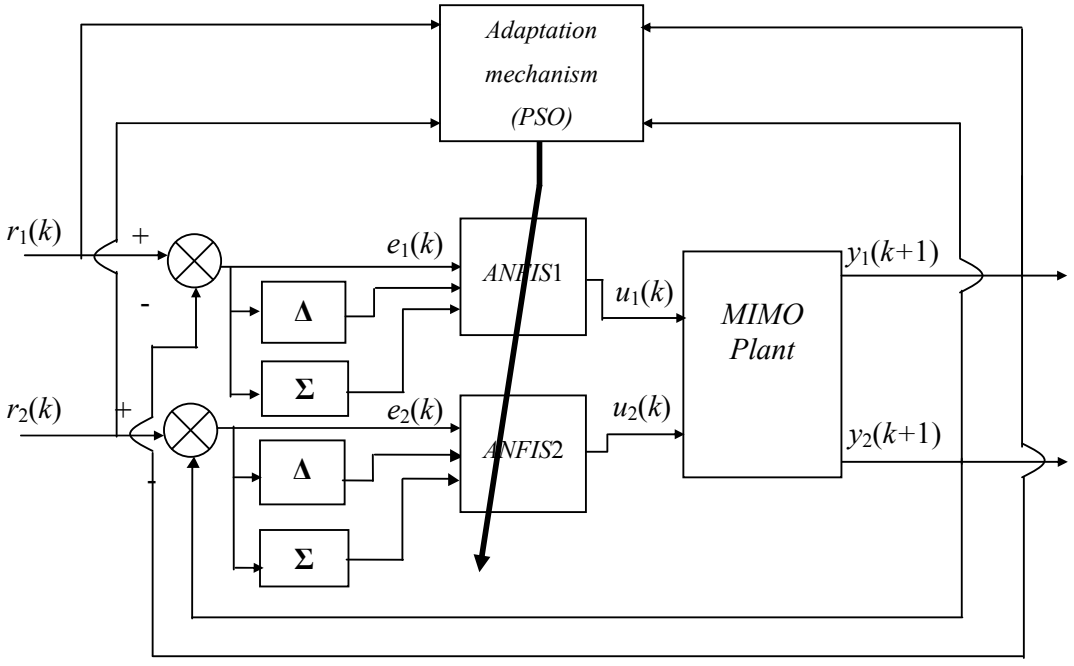
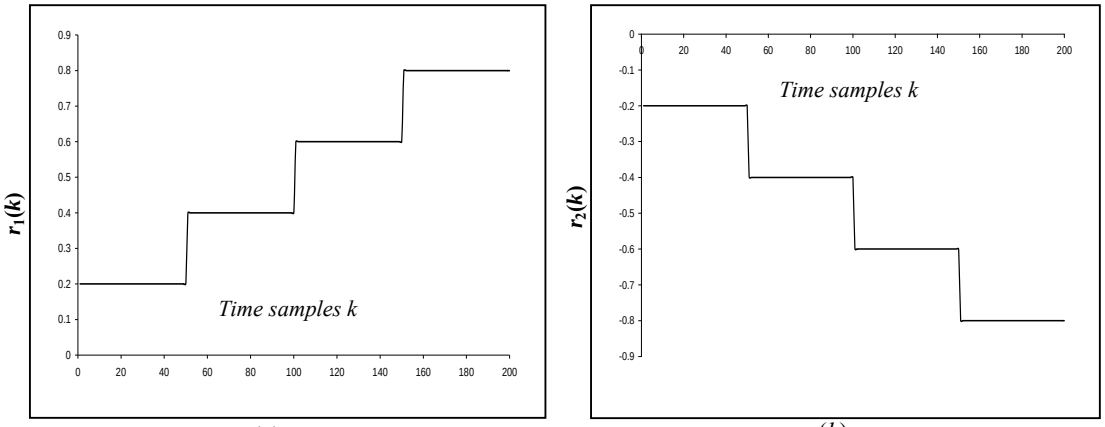
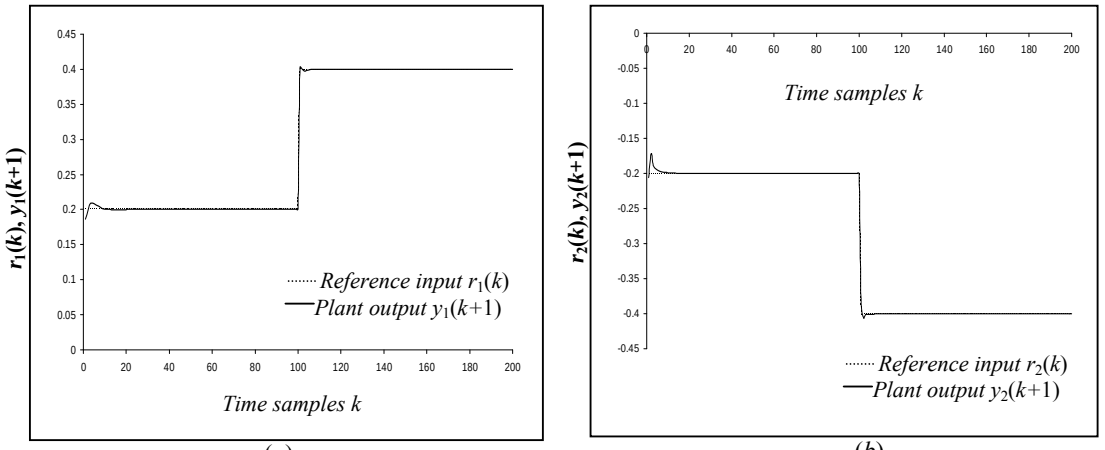


Fig. 2: A MIMO (2×2) plant controlled by the MIMO PID-like ANFIS controller.

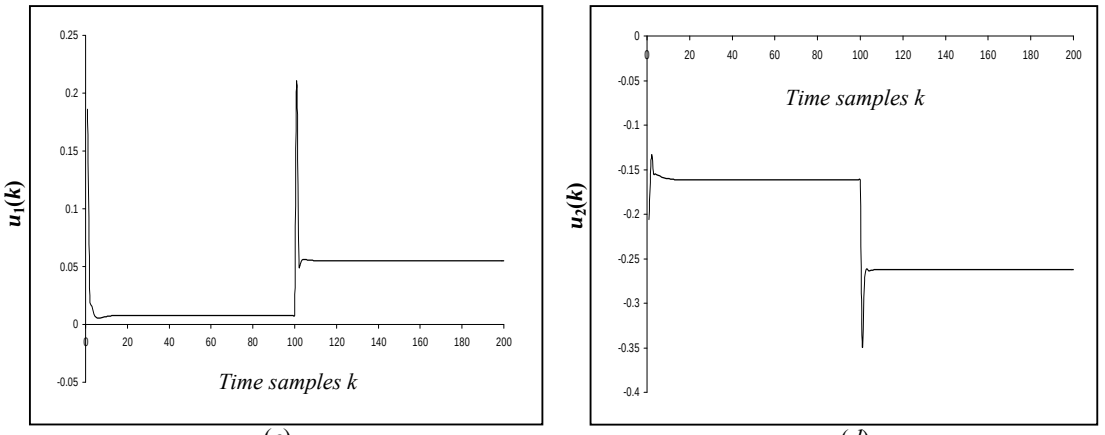


(*a*) (*b*)

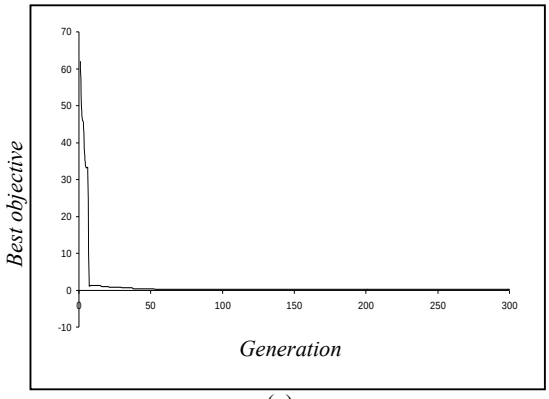
Fig. 3: Training signals (a) for the first input (b) for the second input.



(*a*) (*b*)

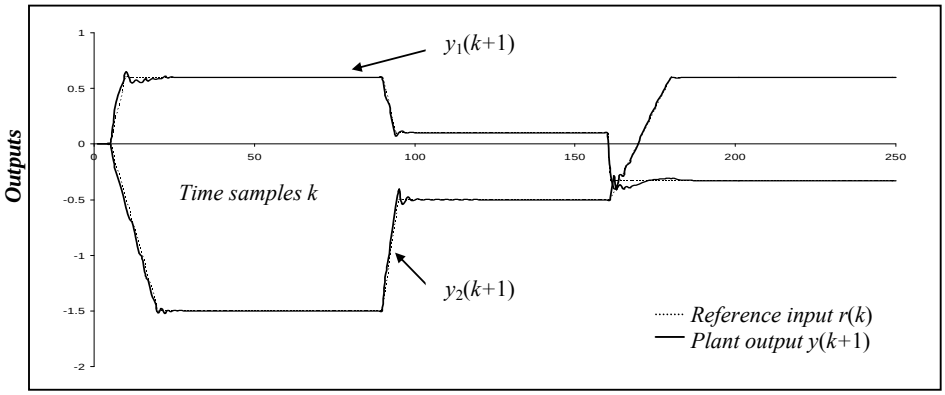


(*c*) (*d*)

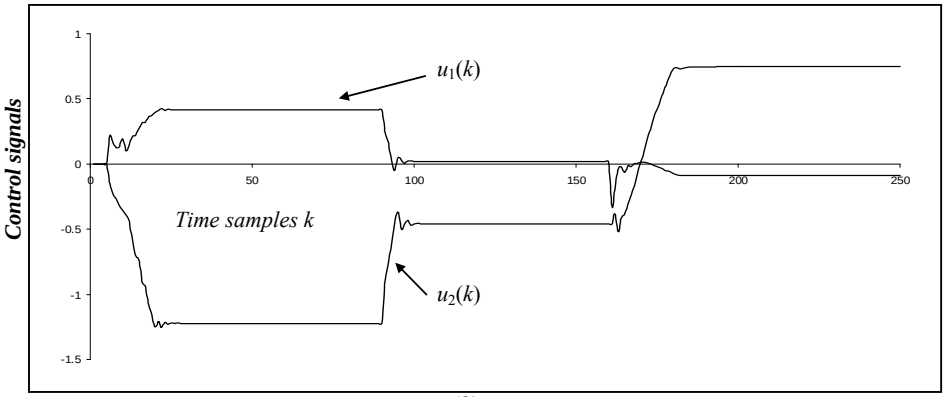


(*e*)

Fig. 4: Plant 1 (a) first reference input and plant output (b) second reference input and plant output (c) first control signal (d) second control signal (e) best 0.5ISE.



(*a*)



(*b*)

Fig. 5: Plant 1 subjected to other reference signals (a) first and second reference inputs and plant outputs (b) first and second control signals.

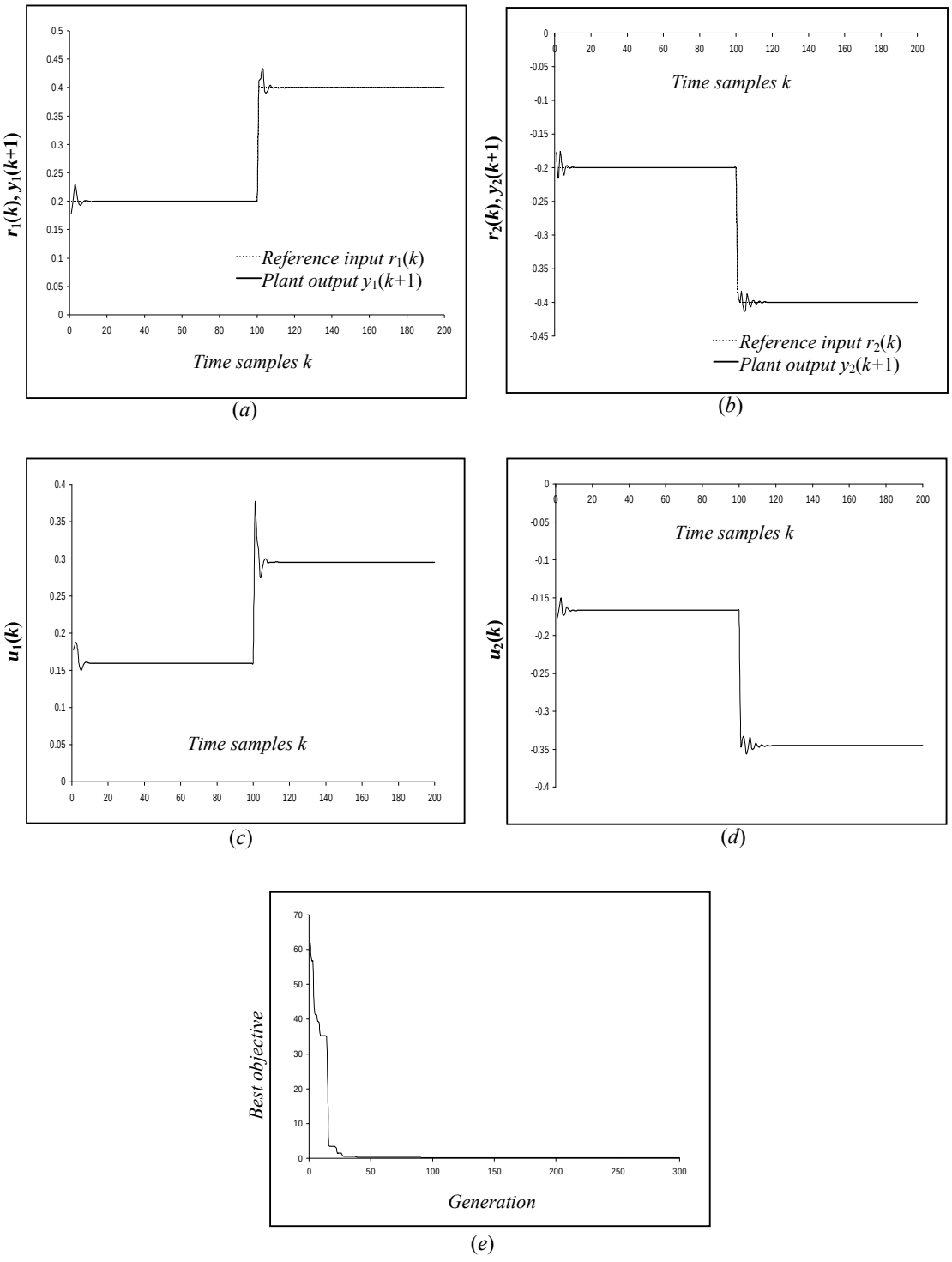


Fig. 6: Plant 2 (a) first reference input and plant output (b) second reference input and plant output (c) first control signal (d) second control signal (e) best 0.5ISE.

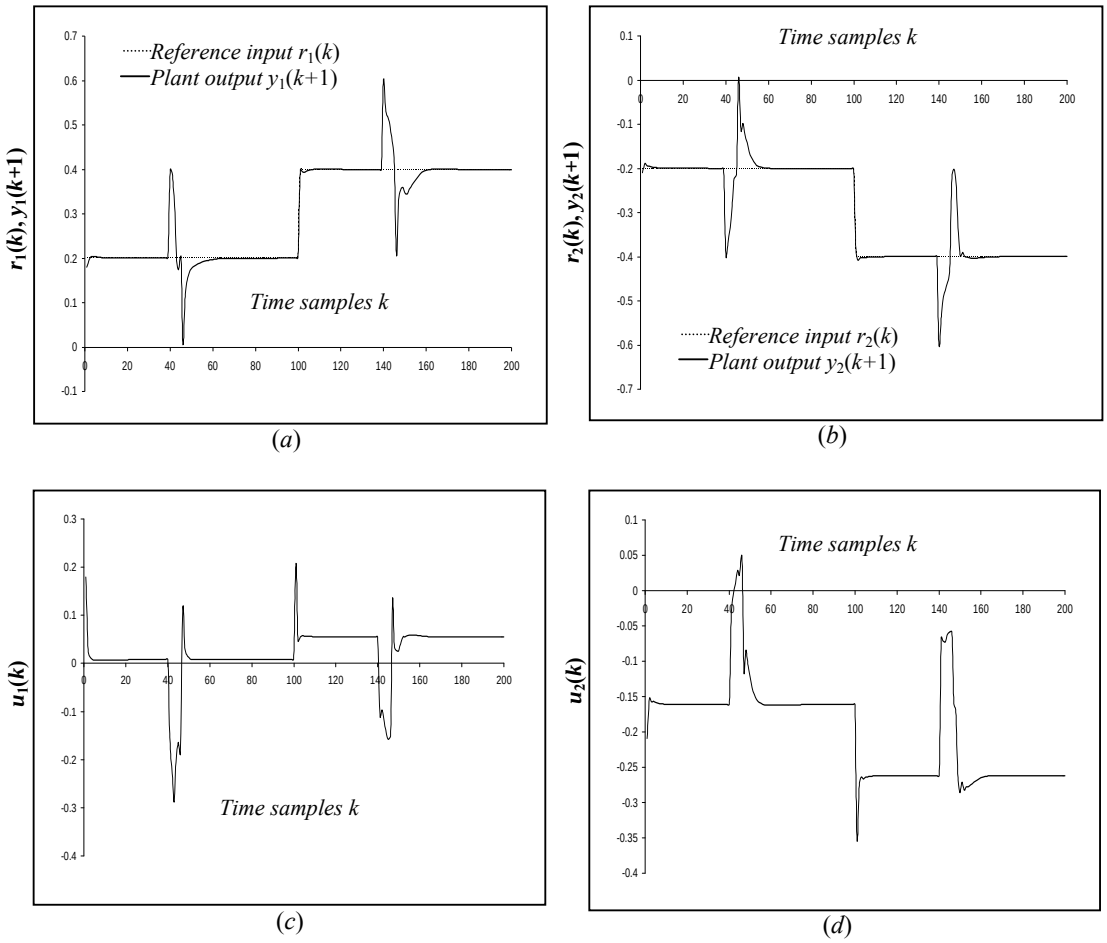


Fig. 7: Plant 1 subjected to 20% external disturbances (a) first reference input and plant output (b) second reference input and plant output (c) first control signal (d) second control signal.

Table 1: Comparison results of the PSO, the GHS, and the GA in the task of training the MIMO ANFIS controller

Optimization Method	criterion	Controlled Plant	
		Plant 1	Plant 2
PSO	Average Training 0.5ISE	0.1638	0.1630
	Average Testing 0.5ISE	0.0815	0.0815
	Average Time (sec.)	135.496	132.668
GHS	Average Training 0.5ISE	0.1728	0.1699
	Average Testing 0.5ISE	0.0884	0.0843
	Average Time (sec.)	200.598	199.266
GA	Average Training 0.5ISE	0.1739	0.1670
	Average Testing 0.5ISE	0.1047	0.0845
	Average Time (sec.)	277.881	281.562

Robustness Test:

This test is done to establish how robust the MIMO ANFIS controller is to environmental changes that might encounter the control system. This test was achieved on Plant 1 by applying a bounded external disturbance of 20% of the first output and –20% of the second output at the testing phase. The first disturbance is applied at the interval ($40 \le k \le 45$) while the second one is applied at the interval ($140 \le k \le 145$) of the two testing signals. Fig. 7 shows the two plant outputs along with the two control actions of the controller for this test. By examining Fig.s 7 (c) and 7 (d), it can be seen that the control actions of the ANFIS controller are adapted to eliminate the effect of the external disturbances, where the convergence to the desired responses is achieved after the adaptation of the control signals as shown in Fig.s 7 (a) and 7 (b). This result gives an indication that this intelligent controller, trained by the PSO technique, has the ability to handle external disturbances.

A Comparative Study with Other Optimization Techniques:

In order to assess the effectiveness of the PSO technique in training the MIMO ANFIS controller, a comparative study is made in this section with a real-coded GA and a newly developed optimization method, namely the GHS. Similar to the PSO, the real-coded GA and the GHS are used to train the MIMO ANFIS controller to control the two nonlinear MIMO plants described before. The same training and testing signals, used for the two previous plants, are used for all the tests. The GHS parameters are set as follows; *HMS*: 50, maximum number of improvisations: 15000, *HMCR*: 0.8, and *PAR*: 0.5. On the other hand, the parameters of the GA are set to the following values; population size: 50, maximum number of generations: 300, P_c: 0.8, and P_m: 0.05. For the current application, and after several simulation tests, these parameter settings for each of the GHS and the GA were found to be adequate to achieve a satisfactory performance in training the MIMO ANFIS controller. Unlike the PSO and the GA, the maximum number of improvisations is set to 15000 in the GHS for the following reason. Since the procedure for both the PSO and the GA involves evaluating the entire group (population) in one iteration (generation), then the total number of performance evaluations in one run will be 50 [group (population) size] × 300 [maximum number of iterations (generations)] = 15000 evaluations. On the other hand, the GHS procedure is designed to evaluate only one solution vector in a given iteration, therefore, the maximum number of improvisations is set to 15000 in the GHS to be equivalent to the 15000 performance evaluations in both the PSO and the GA. Due to the stochastic nature of all the optimization methods involved in this study, the result obtained from one simulation run might be different from that of other runs. For this reason, and in order to achieve a more reliable comparison study, five runs were conducted for each method in controlling each of the two plants. Then the average objective function and average time from each of these five runs are considered as the basis in this comparison. Table 1 summarizes the results of comparing the performances of the PSO, the GHS, and the GA in training the MIMO ANFIS controller.

A careful examination to the results in Table 1 reveals that the PSO technique has achieved the best performance compared to the GHS and the GA for the two plants. In terms of processing time, it is obvious that the PSO has required the least processing time. Unsurprisingly, this performance for the PSO can be attributed to its simple concept and uncomplicated operators. On the other hand, in terms of control accuracy, the PSO has achieved less training and testing 0.5ISE compared to the GHS and the GA for the two plants. Therefore, the results in Table 1 clearly indicate that the PSO technique is a more suitable candidate to train the MIMO ANFIS controller in the task of controlling nonlinear MIMO systems.

Conclusions:

In this paper, the PSO technique has been utilized to train a PID-like ANFIS controller to control nonlinear MIMO systems. Unlike the hybrid learning method, which is widely used to train the ANFIS network, the PSO technique does not require a teaching signal and hence, it is a more suitable method to train the ANFIS network as a feedback controller. Moreover, the trial and error method in finding the optimal settings for the controller scaling factors was avoided by leaving this task to the PSO technique to find these settings. In order to reduce the number of parameters to be optimized by the PSO in the ANFIS structure, only two parameters have been used for the bell-shaped membership functions in the premise part of each rule. In addition, the zero-order Sugeno fuzzy model was used in order to reduce the number of the consequent parameters from 1000 parameters in the case of a first-order Sugeno fuzzy model into 250 parameters in the case of the zero-order model. The simulation results showed the effectiveness of the PSO technique in training the MIMO ANFIS controller in terms of control accuracy and generalization ability. Moreover, from the robustness test, this controller has shown a remarkable ability in eliminating the effects of external disturbances. Finally, based on a comparative study, the PSO technique has shown its superiority over the GHS and the GA in training the MIMO ANFIS controller in terms of control accuracy and processing time.

REFERENCES

Al-Said, I.A.M., 2000. Genetic algorithms based intelligent control, Ph.D. Thesis, University of Technology, Baghdad-Iraq.

Belikov, J. and E. Petlenkov, 2008. Calculation of the Control Signal in MIMO NN-based ANARX Models: Analytical Approach. Proceedings of the 10th International Conference on Control, Automation, Robotics and Vision, Hanoi-Vietnam, pp: 2196-2201.

Cao, H., G. Si, Y. Zhang and X. Ma, 2007. A Hybrid Controller of Self-optimizing Algorithm and ANFIS for Ball Mill Pulverizing System. Proceedings of the IEEE International Conference on Mechatronics and Automation, Harbin-China, pp: 3289-3294.

Chopra, S., R. Mitra and V. Kumar, 2007. Neural Network Tuned Fuzzy Controller for MIMO System. International Journal of Computer Systems Science and Engineering, 2(1): 78-85.

Coelho, L.D.S. and D.L.A. Bernert, 2009. An Improved Harmony Search Algorithm for Synchronization of Discrete-time Chaotic Systems. Chaos, Solitons and Fractals, 41(5): 2526-2532.

Djukanović, M.B., M.S. Ćalović, B.V. Vešović and D.J. Šobajić, 1997. Neuro-fuzzy Controller of Low Head Hydropower Plants Using Adaptive-network Based Fuzzy Inference System. IEEE Transactions on Energy Conversion, 12(4): 375-381.

Gaing, Z.-L., 2004. A Particle Swarm Optimization Approach for Optimum Design of PID Controller in AVR System. IEEE Transactions on Energy Conversion, 19(2): 384-391.

Geem, Z.W., J.H. Kim and G.V. Loganathan, 2001. A New Heuristic Optimization Algorithm: Harmony Search. Simulation, 76(2): 60-68.

Ghomsheh, V.S., M.A. Shoorehdeli and M. Teshnehlab, 2007. Training ANFIS Structure with Modified PSO Algorithm. Mediterranean Conference on Control and Automation, Athens-Greece, pp: 1-6.

Jang, J.-S.R., 1993. ANFIS: Adaptive-network-based Fuzzy Inference System. IEEE Transactions on Systems, Man, and Cybernetics, 23(3): 665-685.

Jang, J.-S.R., C.-T. Sun and E. Mizutani, 1997. Neuro-fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence. Prentice-Hall, Englewood Cliffs, New Jersey.

Kennedy, J. and R. Eberhart, 1995. Particle Swarm Optimization. Proceeding of the IEEE International Conference on Neural Networks, Perth-Australia, pp: 1942-1948.

Lasheen, A.A., A.M. El-Garhy, E.M. Saad and S.M. Eid, 2009. Using Hybrid Genetic and Nelder-Mead Algorithm for Decoupling of MIMO Systems with Application on Two Coupled Distillation Columns Process. International Journal of Mathematics and Computers in Simulation, 3(3): 146-157.

Lin, C.-J., 2004. A GA-based Neural Fuzzy System for Temperature Control. Fuzzy Sets and Systems, 143(2): 311-333.

Linkens, D.A. and H.O. Nyongesa, 1996. Learning Systems in Intelligent Control: An Appraisal of Fuzzy, Neural and Genetic Algorithm Control Applications. IEE proceedings- Control Theory Applications, 143(4): 367-38.

Lutfy, O.F., S.B. Noor, M.H. Marhaban and K.A. Abbas, 2009. A Genetically Trained Adaptive Neuro-fuzzy Inference System Network Utilized as a Proportional-integral-derivative-like Feedback Controller for Non-linear Systems. Proceedings of the Institution of Mechanical Engineers. Part I: Journal of Systems and Control Engineering, 223(3): 309-321.

Mahmoud, T.S., M.H. Marhaban, T.S. Hong, and S. Ng, 2008. ANFIS Controller with Fuzzy Subtractive Clustering Method to Reduce Coupling Effects in Twin Rotor MIMO System (TRMS) with Less Memory and Time Usage. International Conference on Advanced Computer Control, Singapore, pp: 19-23.

Mahmoud, T.S., M.H. Marhaban and T.S. Hong, 2010. ANFIS: Self-tuning Fuzzy PD Controller for Twin Rotor MIMO System. IEEJ Transactions on Electrical and Electronic Engineering, 5(3): 369-371.

Narendra, K.S. and K. Parthasarathy, 1990. Identification and Control of Dynamical Systems Using Neural Networks. IEEE Transactions on Neural Networks, 1(1): 4-27.

Omran, M.G.H. and M. Mahdavi, 2008. Global-best Harmony Search. Applied Mathematics and Computation, 198(2): 643-656.

Petlenkov, E., 2007. NN-ANARX Structure Based Dynamic Output Feedback Linearization for Control of Nonlinear MIMO Systems. Proceedings of the 15th Mediterranean Conference on Control and Automation, Athens-Greece, pp: 1-6.

Sivanandam, S.N. and S.N. Deepa, 2008. Introduction to Genetic Algorithms. Springer-Verlag, Berlin, Heidelberg.

Song, F. and P. Li, 2006. MIMO Decoupling Control Based on Support Vector Machines αth-order Inversion. Proceedings of the 6th World Congress on Intelligent Control and Automation, Dalian-China, pp: 1002-1006.

Song, Y., Z. Chen and Z. Yuan, 2007. New Chaotic PSO-based Neural Network Predictive Control for Nonlinear Process. IEEE Transactions on Neural Networks, 18(2): 595-600.

Toha, S.F. and M.O. Tokhi, 2009. Dynamic Nonlinear Inverse-model Based Control of a Twin Rotor System Using Adaptive Neuro-fuzzy Inference System. Third UKSim European Symposium on Computer Modeling and Simulation, Athens-Greece, pp: 107-111.

Touati, Y., K. Djouani and Y. Amirat, 2002. Neuro-fuzzy Based Approach for Hybrid Force/Position Robot Control. Proceedings of the IEEE International Conference on Industrial Technology, Bangkok-Thailand, pp: 376-381.

Yao, J. and Y. Chai, 2007. Swing-up Control of Double Inverted Pendulum Based on Adaptive Neuro-fuzzy Inference System. Fourth International Conference on Fuzzy Systems and Knowledge Discovery, Haikou, Hainan-China, pp: 195-198.

Zarei, O., M. Fesanghary, B. Farshi, R.J. Saffar and M.R. Razfar, 2009. Optimization of Multi-pass Face-milling via Harmony Search Algorithm. Journal of Materials Processing Technology, 209(5): 2386-2392.

Zhao, L., Y. Yang and Y. Zeng, 2009. Eliciting Compact T-S Fuzzy Models Using Subtractive Clustering and Coevolutionary Particle Swarm Optimization. Neurocomputing, 72(10-12): 2569-2575.

Zhou, C. and K. Jagannathan, 1996. Adaptive Network Based Fuzzy Control of a Dynamic Biped Walking Robot. Proceedings of the IEEE International Joint Symposia on Intelligence and Systems, Maryland-USA, pp: 109-116.