

Neuro-Fuzzy System Design Using Differential Evolution with Local Information

Chin-Teng Lin^{1,2}, Ming-Feng Han¹, Yang-Yin Lin¹, Shih-Hui Liao¹, Jyh-Yeong Chang¹

¹Department of Electrical and Control Engineering, National Chiao Tung University, Taiwan

²Brain Research Center and Department of Electrical Engineering

ctlin@mail.nctu.edu.tw, u9712828.ece97g@nctu.edu.tw, oliver.vylin@gmail.com, lia0622@gmail.com,
jychang@mail.nctu.edu.tw

Abstract—This paper proposes a differential evolution with local information for TSK-type neuro-fuzzy system optimization. The differential evolution with local information consider neighborhood between each individual to keep the diversity of population. An adaptive parameter tuning based on 1/5th rule is used to trade off between local search and global search. For structure learning algorithm, the on-line clustering algorithm is used for rule generation. The structure learning algorithm generates a new rule which compares the firing strength. Initially, there is no rule in neuro-fuzzy system model. The rules are automatically generated by fuzzy measure. For parameter learning, the parameters are optimized by differential evolution algorithm. Finally, the proposed neuro-fuzzy system with novel differential evolution model is applied in chaotic sequence prediction problem. Results of this paper demonstrate the effectiveness of the proposed model.

Keywords— Evolution Algorithm; Neuro-Fuzzy System; Fuzzy System; Differential Evolution Optimization.

I. INTRODUCTION

To train the parameters in designing a neuro-fuzzy system (NFS), many papers had used Backpropagation (BP). The BP is a powerful training technique that can be applied to networks with a forward structure. Since the steepest descent approach is used in BP training to minimize the error function, the algorithms may reach the local minima very quickly and never find the global solution. The aforementioned disadvantages lead to suboptimal performance, even with a favorable neural fuzzy network topology. One major optimization category as evolutionary algorithm (EA) [1-3] for global solution in NFS has been proposed. These algorithms are heuristic and stochastic. They involve populations of individuals with a specific behavior similar to a biological phenomenon, like crossover or mutation. Genetic NFS [4-7] is a well-known case which design FSs using genetic algorithms (GAs).

Another evolutionary algorithms category for FS design, called particle swarm optimization (PSO), appears to be better than the backpropagation algorithm. It is an evolutionary computation technique that was developed by Kennedy and Eberhart in 1995 [8]. The underlying motivation for the development of PSO algorithm is the social behavior of animals, such as bird flocking, fish schooling and swarm theory. PSO has been successfully applied to many optimization problems, such as feedforward neural network

design [9-11] in control problems, with improved performance over GAs.

A differential evolution algorithm (DE), similar to GA, has emerged as a robust numerical optimization algorithm and has been successfully applied to solve various difficult optimization problems [12-14]. The basic strategy employs the difference of two randomly selected individuals as the source of random variations for a third individual. In [15-16], the researcher has proposed a modified differential evolution (MODE) for an adaptive neural fuzzy network (ANFN-MODE). This MODE provides a cluster-based mutation scheme to prevent the algorithm from being trapped in local optima of the search space. However, a DE that lacks the diversity and adaptive parameter tuning is often to affect the performance[14].

This study proposes DE using local information (DELI) for TSK-type NFS optimization. The DELI consider neighborhood between each individual to keep the diversity of population. An adaptive parameter tuning based on 1/5th rule is used to tradeoff between local search and global search. An online learning algorithm, consisting of structure learning and parameter learning, is proposed to construct the NFS automatically. The structure learning algorithm generates a new rule which compares the fire strength. Initially, there is no rule in NFS model. The rules are automatically generated by fuzzy measure. The parameter learning algorithm is based on DELI to tune the free parameters in the NFS. To investigate the effectiveness of the proposed NFS-DELI, it is applied to dynamic system identification. The proposed NFS-DE model shows effectiveness results in simulations.

This paper is organized as follows. Section II describes the basic structure and function for NFS. Section III introduces the rule generation and parameter optimization algorithm in DELI. Section IV presents NFS design results of DELI and its performance comparisons with other evolutionary algorithms. Finally, Section V draws conclusions.

II. THE STRUCTURE OF NFS

A four-layered NFS is shown in Fig. 1, which is comprised of the input, membership function, rule, and output layers. The signal propagation and the basic functions in each layer are described as follows.

Layer 1—Input layer: No computation is done in this layer. Each node in this layer, which corresponds to one input

variable, only transmits input values to the next layer directly. That is

$$O^{(1)} = z_i. \quad (1)$$

Where $i=1,2,\dots,n$ are the input variables of the NFS.

Layer 2—Membership function layer: Each node in this layer is a membership function that corresponds one linguistic label of one of the input variables in Layer 1. In other words, the membership value which specifies the degree to which an input value belongs to a fuzzy set is calculated in Layer 2

$$O^{(2)} = \mu_{ij} = \exp\left(\frac{-(z_i - m_{ij})^2}{\sigma_{ij}^2}\right). \quad (2)$$

Where $i=1,2,\dots,n$, $j=1,2,\dots,M$, m_{ij} and σ_{ij} are respectively, the center (or mean) and the width (or variance) of the Gaussian membership function of input variable. M is number of rules.

Layer 3—Rule layer: A node in this layer represents a fuzzy logic rule and performs precondition matching of a rule. Here, we use the AND operation for each Layer 2 node

$$O^{(3)} = R_j = \prod_{i=1}^n \mu_{ij}. \quad (3)$$

The output of a layer 3 node represents the firing strength of the corresponding fuzzy rule.

Layer 4—Output layer: Each node in this layer corresponds to one output variable. The node integrates all of the actions recommended by layers 4 and acts as a defuzzifier with

$$O^{(4)} = y = \frac{\sum_{j=1}^M R_j \left(\sum_{i=1}^l s_{ij} z_i + s_{0j} \right)}{\sum_{j=1}^M R_j}. \quad (4)$$

In this paper, the consequent part use a TSK type which linear combination of TSK parameters s_{ij} and input variables.

III. LEARNING ALGORITHMS OF THE NFS

This section describes the basic concepts of two types of learning algorithm. Section III-A describes the rule generation based on-line clustering algorithm for structure learning. Section III-B describes the DELI algorithm for parameter optimization.

A. Using on-line clustering algorithm for rule generation

The first task in structure learning is to decide when to generate a new rule. In many study, an on-line cluster algorithm is used to neuro-fuzzy system for generating fuzzy rule [2-5]. Fig.2. shows a detailed flow chart for this online procedure. Initially, there is no rule in NFS. For first incoming data, NFS generates a first fuzzy rule. The mean of membership function set the value of first incoming data, the variance of membership function set initial value. The weights set value randomly between the ranges. After first incoming data, the firing strength is calculated by Layer 3. A maximum

firing strength in Layer 3 has to compare with a threshold which pre-specified threshold between 0 and 1. If the maximum firing strength is smaller than threshold, then a new rule is generated. Next step is to assign the value of mean, variance, and weights for the new rule. Repeating the above process for every incoming piece of training data generates new rules, until the iteration is termination.

B. Differential evolution with local information (DELI)

The DELI learning algorithm consists of three major phases: the initialization phase, the DE operator phase, and the parameter tuning phase.

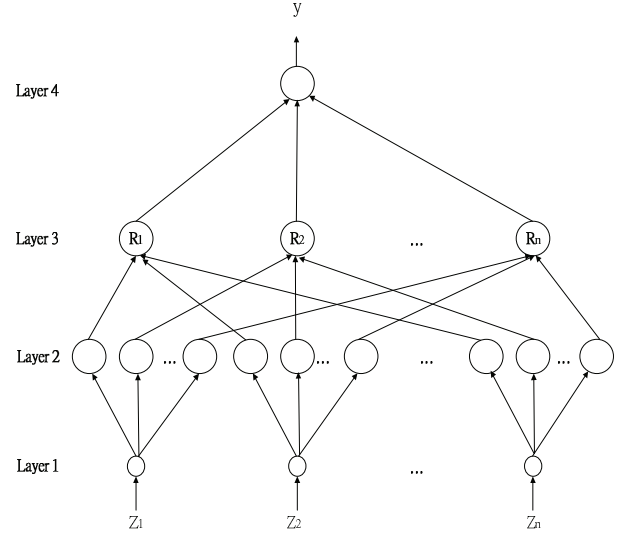


Fig. 1. Architecture of the NFS.

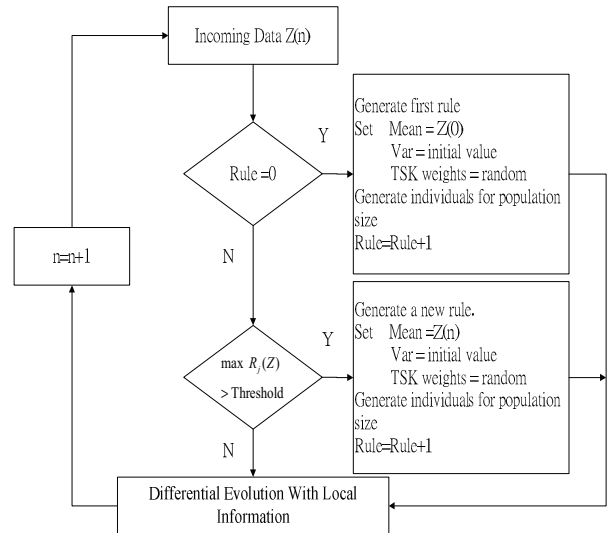


Fig. 2. The flow chart for on-line procedure.

In the initialization phase, every individual is consisted of mean, variance, and TSK weights. An individual is a neuro-fuzzy system in this paper. The detailed individual structure is shown in Fig.2. Before the DELI learning algorithm is applied, every individual must be created randomly. For evaluation of

performance, we adopt a fitness function to evaluate the performance of each individual as follows:

$$fitness = \sqrt{\frac{1}{N} \sum_{i=1}^N (y_i - \bar{y}_i)^2}, \quad (5)$$

where y_i represents the model output of the i th data, $\bar{y}_i$ the desired output of the i th data, and N the number of the training data.

For DE operator phase, first operation is mutation operator. A mutated individual is generated according to the following equation:

$$v_g = (1 - \alpha)(x_1 + (1 - F)(x_2 - x_3) + F(x_{local} - x_1)) + \alpha(x_1 + (1 - F)(x_2 - x_3) + F(x_{gbest} - x_1)) \quad (6)$$

where F , commonly known as the scaling factor, α is a control factor, x_{gbest} is the best fitness value up to now in the population and x_{local} is the local best fitness value that consider the neighborhood of individual. Three individuals (x_1 , x_2 , and x_3) are randomly selected in population for operation.

Second operation is crossover operator which mutated individual v_i and random individual x_i generate the offspring. If a random number smaller than crossover probability, the mutated individual v_i is selected for the element of offspring. Else, random individual x_i is selected. Finally, better individuals based on fitness function values are reserved to the next generation.

To prevent the DE from being trapped in the local optima of the search space, an adaptive parameter tuning is used in this paper. This scheme uses 1/5th rule to increase the search capability. The basic concept of 1/5th rule is derived from the Evolution Strategies [17-18]. The short description of the 1/5th rule is present in Figure 3.

1. perform the DELI for a number G of generations:
 - keep F constant during this period.
 - count the number N of better G_{best} during this period.

2. determine an estimate of success probability P

$$P = N/G$$

3. change F according to

$$F = \begin{cases} F / \beta & \text{if } P > 1/5 \\ F \cdot \beta & \text{if } P < 1/5 \\ F & \text{if } P = 1/5 \end{cases}, 0 < \beta < 1$$

Fig. 3. The 1/5th code in DELI.

IV. SIMULATION RESULTS

The time-series prediction problem used in this example is the chaotic Mackey–Glass time series, which is generated from the following differential equation:

$$\frac{dx(t)}{dt} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \quad (7)$$

where $\tau > 17$. In this paper, the parameter $\tau = 30$, and $x(0) = 1.2$. Four past values are used to predict $x(t)$, and the input–output data format is given by

$$[x(t-24), x(t-18), x(t-12), x(t-6) | x(t)]$$

A total of 1000 patterns are generated from $t = 124$ to 1123, where the first 500 patterns are used to train, and the last 500 are used to test. A total of 50 runs are performed for statistical analysis. As in the previous study [11], a first-order TSK-type FS is used. In DELI, the threshold μ_{th} is set to 0.011 α is set to 0.4, β is set to 0.9. Fig. 4 shows the average best-so-far training RMSE for each evaluation. DELI have better fitness value and learning capability. Table 1 shows the average training and test RMSEs over 50 runs. Comparison of GA, PSO, DE, MODE[15], HGAPSO[11], DELI, the proposed DELI have a good performance. Fig. 5 shows the desired output (blue line) and actual output (red line) of DELI.

The result shows the best prediction output of DELI.

V. CONCLUSIONS

This paper proposed a DELI for dynamic system identification. The DELI is simple evolution algorithm with neighborhood information, and keeps the diversity in population. DELI with 1/5th rule parameter tuning scheme tradeoff between local search and global search during evolution to effectively find a good solution for parameter optimization. An online cluster algorithm is used for rule generation. The experimental results demonstrate that the DELI obtains a smaller RMSE value than other algorithm for dynamic system identification.

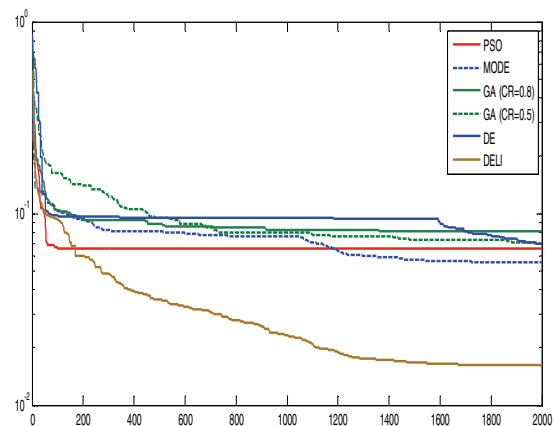


Fig. 4. Training RMSEs at each performance evaluation for DELI and other algorithms.

Table1. Performance of the DELI and other algorithms for the chaotic sequence prediction problem.

	GA	PSO	DE	HGAPSO[11]	MODE[15]	DELI
Rule number	4	4	4	4	4	4
Training RMSE	0.081	0.066	0.069	0.035	0.041	0.016
Testing RMSE	0.090	0.077	0.072	0.047	0.053	0.020

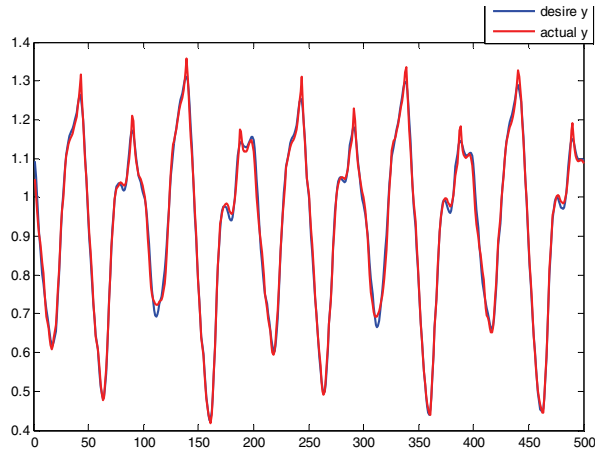


Fig. 5. Results of the DELI-optimized NFS in simulation.

REFERENCES

- [1] C. T. Lin and C. S. G. Lee, *Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent System*. Englewood Cliffs, NJ: Prentice-Hall, 1996
- [2] O. Cordon, F. Herrera, F. Hoffmann, and L. Magdalena, *Genetic Fuzzy Systems*. Singapore: World Scientific, 2001.
- [3] E. Sanchez, T. Shibata, and L. A. Zadeh, *Genetic Algorithms and Fuzzy Logic Systems: Soft Computing Perspectives*. Singapore: World Scientific, 1997.
- [4] C. F. Juang, "A TSK-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithms," *IEEE Trans. Fuzzy Syst.*, vol. 10, no. 2, pp. 155–170, Apr. 2002.
- [5] C. F. Juang, "Combination of on-line clustering and Q-value based GA for reinforcement fuzzy system design," *IEEE Trans. Fuzzy Syst.*, vol. 13, no. 3, pp. 289–302, Jun. 2005.
- [6] C. H. Chou, "Genetic algorithm-based optimal fuzzy controller design in the linguistic space," *IEEE Trans. Fuzzy Syst.*, vol. 14, no. 3, pp. 372–385, Jun. 2006.
- [7] F. Hoffmann, D. Schauten, and S. Holemann, "Incremental evolutionary design of TSK fuzzy controllers," *IEEE Trans. Fuzzy Syst.*, vol. 15, no. 4, pp. 563–577, Aug. 2007.
- [8] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, Dec., 1995, pp. 1942–1948.
- [9] C. F. Juang, "A hybrid of genetic algorithm and particle swarm optimization for recurrent network design," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 2, pp. 997–1006, Apr. 2004.
- [10] K. D. Sharma, A. Chatterjee, and A. Rakshit, "A hybrid approach for design of stable adaptive fuzzy controllers employing Lyapunov theory and particle swarm optimization," *IEEE Trans. Fuzzy Syst.*, vol. 17, no. 2, pp. 329–342, Apr. 2009.

- [11] C. F. Juang, C. M. Hsiao, and C. H. Hsu, "Hierarchical Cluster-Based Multispecies Particle-Swarm Optimization for Fuzzy-System Optimization," *IEEE Trans. Fuzzy Syst.*, vol. 18, no. 1, pp. 14–26, Feb. 2010.
- [12] R. Storn, "System design by constraint adaptation and differential evolution," *IEEE Trans. Evol. Comput.*, vol. 3, no. 1, pp. 22–34, Apr. 1999.
- [13] K. V. Price, R. M. Storn, and J. A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*. Berlin, Germany: Springer-Verlag, 2005.
- [14] J. Liu and J. Lampinen, "A fuzzy adaptive differential evolution algorithm," *Soft Comput.—A Fusion Found., Methodol. Appl.* vol. 9, no. 6, pp. 448–462, 2005.
- [15] C.H Chen, C.J Lin, and C.T Lin, "Nonlinear System Control Using Adaptive Neural Fuzzy Networks Based on a Modified Differential Evolution" *IEEE Trans. Syst. Man Cybern. Part C*. 2009, pp. 459–473.
- [16] C. T. Lin, and M. F. Han, "Differential Evolution based Optimization of Locally Recurrent Neuro-Fuzzy System for Dynamic System Identification," *The 17th National Conference on Fuzzy Theory and its Applications*, pp. 702–707, 2010.
- [17] Thomas Bäck, T., & Schwefel, H.-P., "Evolution strategies I: Variants and their computational implementation," *Genetic Algorithms in Engineering and Computer Science*, pp. 111–126. 1995.
- [18] Beyer, H.-G., & Schwefel, H.-P., "Evolution strategies: A comprehensive introduction," *Natural Computing*, 3–52, 2002.