

An Immune Symbiotic Evolution Learning for Compensatory Neural Fuzzy Networks and Its Applications

Cheng-Hung Chen, *Member, IEEE*

Department of Electrical Engineering
National Formosa University
Yunlin County 632, Taiwan, R.O.C.
E-mail: chchen.ee@nfu.edu.tw

Cheng-Jian Lin, *Member, IEEE*

Department of Computer Science and
Information Engineering
National Chin-Yi University of Technology
Taichung County 411, Taiwan, R.O.C.
E-mail: cjlin@nctu.edu.tw

Chin-Teng Lin, *Fellow, IEEE*

Department of Electrical Engineering
National Chiao-Tung University
Hsinchu 300, Taiwan, R.O.C.
E-mail: ctlin@mail.nctu.edu.tw

Abstract—This study presents an efficient immune symbiotic evolution learning algorithm for the compensatory neural fuzzy network (CNFN). The proposed immune symbiotic evolution learning method (ISEL) includes three major components - initial population, subgroup symbiotic evolution and immune system algorithm. The advantage of the proposed ISEL method are that the subgroup symbiotic evolution method uses the subgroup-based population to evaluate the fuzzy rules locally and the adopted immune system algorithm can accelerate the search and increase global search capacity. Finally, the simulation results have shown that the proposed CNFN-ISEL can outperform other methods.

Keywords- Compensatory fuzzy operator; neural fuzzy network; symbiotic evolution; immune system algorithm ; control problems.

I. INTRODUCTION

Many researchers have developed GAs to implement fuzzy systems and neural fuzzy networks in order to automate the determination of parameters and structures [1]-[3]. Unfortunately, the number of fuzzy rules (i.e., the length of each chromosome in the GA) increased exponentially as the number of inputs increased. However, such GAs encounter some important issues: 1) the input space is partitioned into a grid; 2) the number of fuzzy rules is determined by trial and error or by a expert system; 3) a fuzzy system was encoded into a chromosome; and 4) the rate of convergence of the evolution process is low.

This study presents an efficient immune symbiotic evolution learning algorithm (ISEL) for the compensatory neural fuzzy network. The proposed compensatory neural fuzzy network (CNFN) is based on our previous research [4]. The immune symbiotic evolution learning method (ISEL) includes three major components - initial population, subgroup symbiotic evolution and immune system algorithm. First, the self-clustering algorithm that determines proper input space partitioning and finds the mean and variance of the membership function and the number of rules, is applied to the initial population. Second, the subgroup symbiotic evolution method in which each sub-antibody represents a single fuzzy rule differs from traditional GAs. A fuzzy system with R -rules is constructed by selecting and combining R sub-antibodies from a given population, and allowing the evolution of the rule itself. Third, the immune system algorithm uses the clonal selection principle to cancel antibodies between others of high similar degree, and these antibodies, after the process, will be

of higher quality, accelerating the search and increasing the global search capacity.

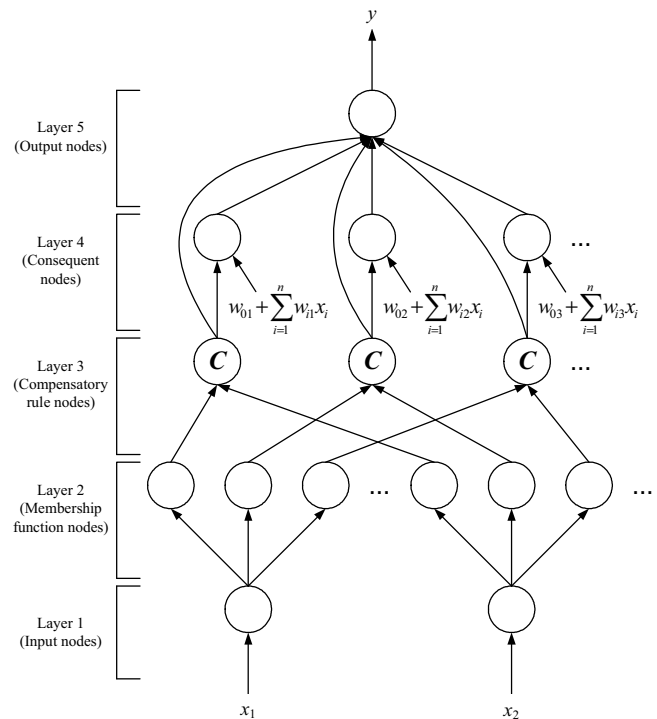


Figure 1. Structure of the proposed CNFN.

II. STRUCTURE OF COMPENSATORY NEURAL FUZZY NETWORK

The section describes the compensatory neural fuzzy network (CNFN) [4]. Compensatory operators are used to optimize fuzzy logic reasoning and to select optimal fuzzy operators. Therefore, an effective neural fuzzy network should be able not only to adaptively adjust fuzzy membership functions but also to dynamically optimize adaptive fuzzy operators. Figure 1 shows the structure of the CNFN.

The CNFN realizes a fuzzy model in the following form.

$$\text{Rule } j: \text{ IF } [x_1 \text{ is } A_{1j} \text{ and } x_2 \text{ is } A_{2j} \dots \text{ and } x_i \text{ is } A_{ij} \dots \text{ and } x_N \text{ is } A_{Nj}]^{1-\gamma_j+\frac{\gamma_j}{N}} \quad (1)$$

$$\text{THEN } y' \text{ is } w_{0j} + w_{1j}x_1 + w_{2j}x_2 + \dots + w_{ij}x_i + \dots + w_{Nj}x_N$$

where x_i is the input variable; y' is the output variable; A_{ij} is the linguistic term of the precondition part; $\gamma_j \in [0,1]$ is the

compensatory degree, and w_{0j} and w_{ij} are the corresponding parameters of consequent part.

III. AN IMMUNE SYMBIOTIC EVOLUTION LEARNING FOR THE CNFN

This section describes an immune symbiotic evolution learning method (ISEL) for constructing the CNFN. The ISEL method comprises of three major components - initial population, subgroup symbiotic evolution and immune system algorithm. First, the initial population uses the self-clustering algorithm that determines proper input space partitioning and finds the mean and variance of the Gaussian membership function and the number of rules. Second, in traditional symbiotic evolution, all rules generally evolve together; however, in the subgroup symbiotic evolution method, each rule evolves separately. Third, the immune system algorithm uses the clonal selection principle to accelerate the search and increase global search capacity. Figure 2 presents the immune symbiotic evolution learning of the CNFN.

A. Initial Population

This subsection introduces the production of initial population, covering the coding and initialization steps. The coding step involves the membership functions and the fuzzy rules of a fuzzy system that represent sub-antibodies and are suitable for subgroup symbiotic evolution. The initialization step assigns the population values before the evolution process begins.

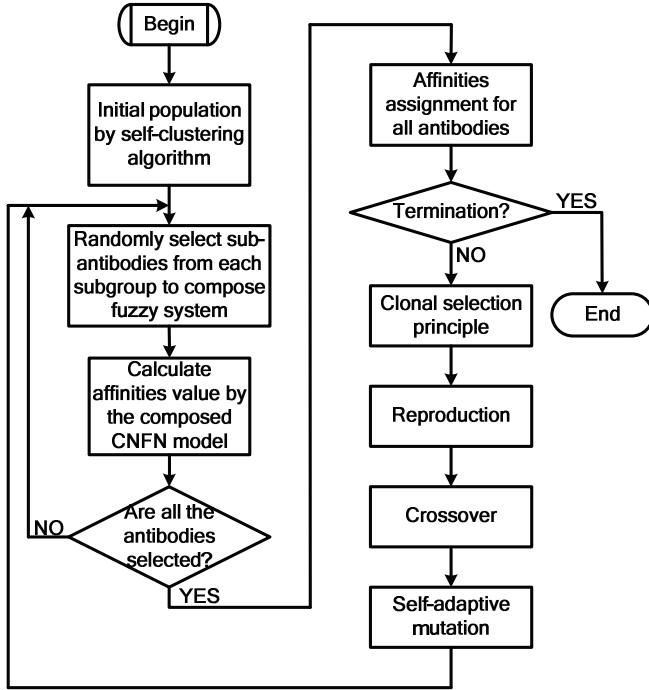


Figure 2. Flowchart of the proposed ISEL method.

1) Coding Step

The first step in ISEL method is the coding of a fuzzy rule into a sub-antibody. Figure 3 shows an example of a fuzzy rule

coded into a sub-antibody where i and j are the i th dimension and the j th rule. Figure 3 describes a fuzzy rule given by Eq. (1), where m_{ij} and σ_{ij} are the mean and variance of a Gaussian membership function, respectively; c_j and d_j are compensatory parameters, and w_{ij} is the corresponding parameter of the consequent part associated with the j th rule node.

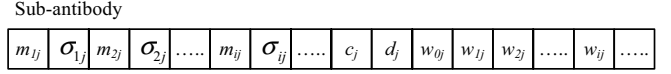


Figure 3. Coding a fuzzy rule into a sub-antibody in subgroup symbiotic evolution.

2) Initialization Step

Before the ISEL method is designed, the individuals that will constitute an initial population must be created. In this study, the initial population was created according to the range of the mean and variance of the membership function, which were computed by the self-clustering algorithm (SCA) method [5]. The following formulations show the generation of the initial population.

Mean: $Sub\text{-}antibody[index] = m_{ij} + random[0,1] \times \sigma_{ij}$
where $index=1,3,...,2 \times N-1$

Variance: $Sub\text{-}antibody[index] = 4 \times random[0,1] \times \sigma_{ij}$
where $index=2,4,...,2 \times N$

Other parameters: $Sub\text{-}antibody[index] = random[-1,1]$
where $index > 2 \times N$

where $index$ is the site of a sub-antibody; m_{ij} and σ_{ij} are the mean and variance, respectively of the Gaussian membership function of the j th rule of the i th input variable. The size of the population depends on the complexity of the problem. Many experiments showed that the population size of 20 is most effective. Besides the population size, other parameters must be set. This is the number of fuzzy systems to be formed and evaluated in each generation. The crossover rate, and the mutation rate and these parameters also depend on the complexity of the problem.

B. Subgroup Symbiotic Evolution

This subsection presents a novel method of subgroup symbiotic evolution. As described above, in the subgroup symbiotic evolution, the affinity value of a rule (a sub-antibody) is computed as the sum of the affinity values of all the feasible combinations of that rule with all other randomly selected rules, and then dividing this sum by the total number of combinations. Figure 4 shows the structure of the sub-antibody in the subgroup symbiotic evolution. The stepwise assignment of the affinity value is as follows.

Step 0: Divide the rules into subgroups of size ps .

Step 1: Randomly select R fuzzy rules (sub-antibody) from each of the above subgroups, and compose the fuzzy system using these R rules.

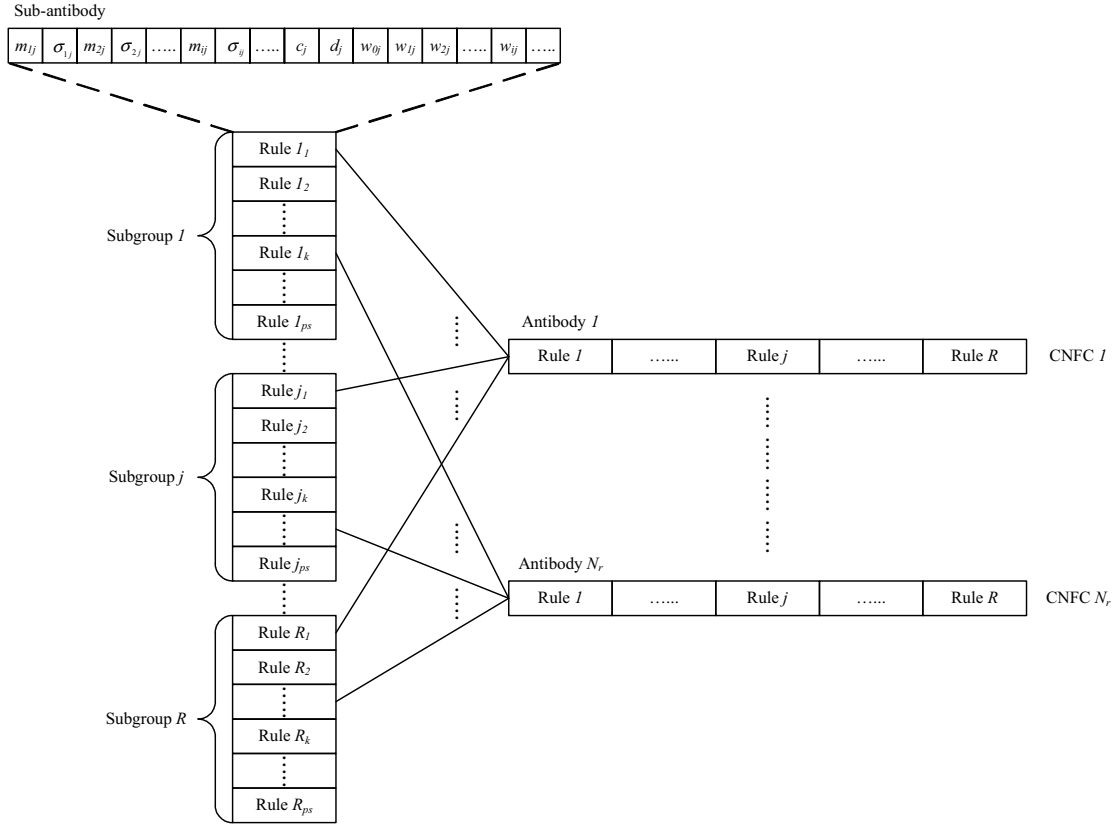


Figure 4. Structure of sub-antibody in subgroup symbiotic evolution.

Step 2: Calculate affinities of the antibodies using the CNFN thus composed. In this study, the affinity value is given by the follow formula;

$$\text{Affinity value} = \frac{1}{1 + \sqrt{\frac{1}{N_t} \sum_{k=1}^{N_t} (y_k - y_k^d)^2}} \quad (2)$$

where y_k represents the k th model output; y_k^d represents the desired output, and N_t represents the number of input data.

Step 3: Divide the affinity value by R and accumulate the divided affinity value to the affinity record of the R selected rules with their recorded affinity values initially set to zero.

Step 4: Repeat the above steps until each rule (sub-antibody) in each subgroup has been selected a sufficient number of times, and record the number of fuzzy systems to which each sub-antibody has contributed.

Step 5: Divide the accumulated affinity of each sub-antibody by the number of times it has been selected.

Step 6: Sort these sub-antibodies in each subgroup in order of decreasing affinity.

C. Immune System Algorithm

This subsection introduces the implementation of the immune system algorithm that is based on the clonal selection principle, the reproduction step, the crossover step and the self-adaptive mutation step.

1) Clonal Selection Principle

The antibodies of this immune system that eliminate the antigen should exhibit at least a minimal threshold of diversity. Accordingly, the diversity operator is applied to these antibodies such that those between others of high similar degree are canceled, guaranteeing and improvement in quality. This process is called clonal selection principle. Therefore, if the affinity between two antibodies exceeds the suppression threshold Th_{affs} then these two antibodies are called similar. Now, the subgroup is assumed to be composed of ps sub-antibodies with L genes, as shown in Fig. 5.

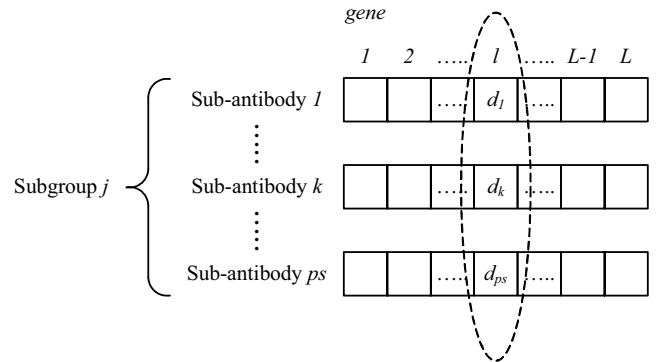


Figure 5. Composition of subgroup with these sub-antibodies.

Information theory yields the entropy value $EV_l(ps)$ of the l th gene in this j th subgroup:

$$EV_l(ps) = \sum_{k=1}^{ps} -P_{kl} \log P_{kl} \quad (3)$$

where P_{kl} is the probability of the k th allele from the l th gene, and $P_{kl} = 1/(1 + \|d_k - d_{k'}\|)$ is defined, where d_k and $d_{k'}$ are the genes of the k th and the k' th sub-antibody, respectively. The diversity of the genes is calculated using Eq. (3). If all alleles of the l th gene are of the same form, then the entropy value $EV(ps)$ will be zero.

The average entropy value $EV(ps)$ of diversity is also given as follows;

$$EV(ps) = \frac{\sum_{l=1}^L EV_l(ps)}{L} \quad (4)$$

where L is the size of the gene in a sub-antibody. Equation (4) yields the diversity of the sub-antibody pool in terms of the entropy. Two affinity expression are considered in the proposed approach. The first explains the relationships between the antibody and the antigen, and is given by Eq. (2). The second describes the degree of association between sub-antibodies. The following equation computes the affinity between sub-antibodies;

$$Affinity_Ab_{kk'} = \frac{1}{1 + EV(2)} \quad (5)$$

where $Affinity_Ab_{kk'}$ is the affinity between two sub-antibodies k and k' , and $EV(2)$ is the entropy of only the sub-antibodies k and k' . This affinity is constrained from zero to one. When $EV(2)$ is zero, the genes of the k th sub-antibody and the k' th sub-antibody are the same. Therefore, the clonal selection principle is that if the $Affinity_Ab_{kk'}$ exceeds Th_{aff} , then the sub-antibody should be canceled. Reproduction, crossover and mutation steps follow, according to the flowchart presented in Fig. 6.

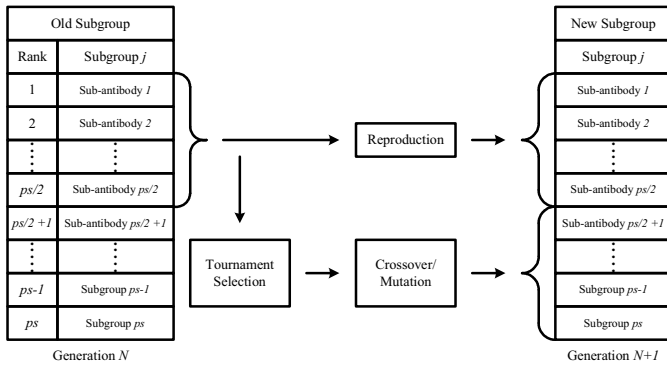


Figure 6. Flowchart of reproduction, crossover and mutation steps.

2) Reproduction Step

Reproduction is a process in which sub-antibody strings are copied according to their affinity values. This process is an artificial version of natural selection. This study uses the roulette-wheel selection method [6] - a simulated roulette is spun - in this reproduction step. The best performing sub-antibodies in the top half of the subgroup evolves to the next generation. The other half is generated to perform crossover on sub-antibodies in the top half of the parent generation.

3) Crossover Step

Reproduction directs the search toward the best existing sub-antibodies but does not create any new sub-antibodies. In nature, an offspring has two parents and inherits genes from both. The main operator working on the parents is the crossover operator, the operation of which occurred for a selected pair with a crossover rate that was set to 0.5 in this study. Therefore, the new sub-antibodies are principally generated by crossover. In this process, the first step is to select two sub-antibodies for crossover. Two sub-antibodies for crossover are selected by tournament selection [6] on the top half of the best-performing sub-antibodies. Tournament selection shown in Fig.7 randomly selects three sub-antibodies in each subgroup, and the sub-antibody with the best affinity is adopted as the designated sub-antibody. The other designated sub-antibody is selected in the same way. Two new sub-antibodies are generated by performing crossover on the selected designated sub-antibodies.

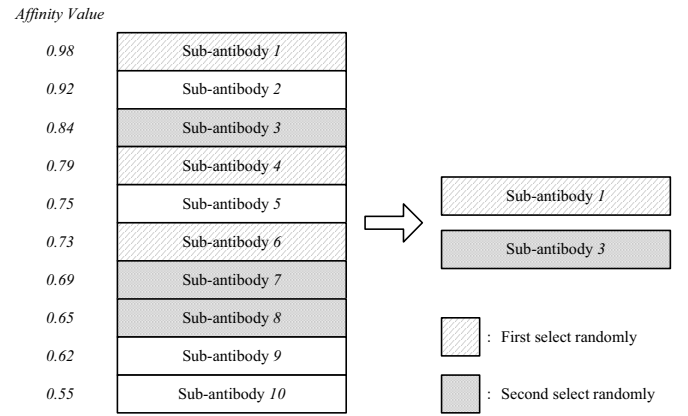


Figure 7. Tournament selection.

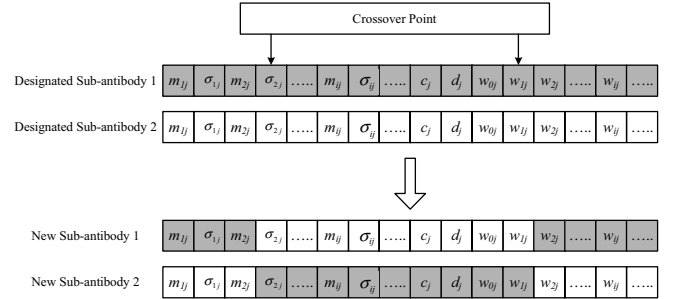


Figure 8. Two-point crossover.

In the second step, the two designated sub-antibodies are crossed and separated using a two-point crossover, as shown in Fig. 8, in which new sub-antibodies are created by exchanging the gene between the selected sites of the two designated sub-antibodies. In the crossover step, we also keep the same number of sub-antibodies in the other half of the population for each subgroup. After this operation, the new sub-antibodies replace the designated sub-antibodies with poor performances.

4) Self-Adaptive Mutation Step

After reproduction and the crossover step yielded new strings, no new information is introduced to the subgroup at the site of a sub-antibody. As a source of new sites, mutation

should be used sparingly because it is a random search operator. In the following simulations, a mutation rate was set to 0.3. Mutation is an operator that randomly alters the allele of a gene. The self-adaptive mutation adopted in the immune system algorithm yields high diversity. The sub-antibody suffers from a mutation to avoid falling in a local optimal solution and to ensure the searching capacity of approximate global optimal solution. The self-adaptive mutation with real-valued representation can be used to produce mutants by adding a number randomly selected from some interval to a given antibody. Figure 9 shows the self-adaptive mutation of a sub-antibody. The mutation value is generated according to Eq. (6). Following self-adaptive mutation, a new sub-antibody can be introduced into the population.

$$SAb[index] = \begin{cases} SAb[index] + (\max(SAb[:]) - SAb[index]) \times \lambda, & \text{if } \alpha > 0.5 \\ SAb[index] + (\min(SAb[:]) - SAb[index]) \times \lambda, & \text{if } \alpha \leq 0.5 \end{cases} \quad (6)$$

where $\lambda = \beta \times (t/T)^\beta$, $\alpha, \beta \in [0,1]$ are random values; $index$ represents the gene in a sub-antibody (SAb); t is the t th generation, and T is the maximum number of generations.

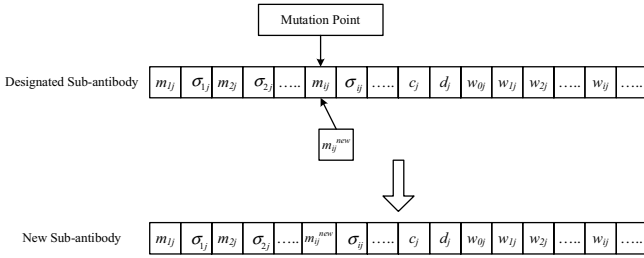


Figure 9. Self-adaptive mutation.

TABLE I. INITIAL PARAMETERS BEFORE TRAINING.

Parameter	Value
Size of Subgroup (ps)	20
Crossover Rate	0.5
Mutation Rate	0.3
Threshold of Affinity (Th_{aff})	0.95
Coding Type	Real Number

IV. ILLUSTRATIVE EXAMPLES

This study evaluated the performance of the proposed compensatory neural fuzzy network using immune symbiotic evolution learning algorithm (CNFN-ISEL) for nonlinear control systems. This section presents several examples and compares the performance with that of other neural fuzzy networks using symbiotic evolution (NFN-SE) [7], [8]. In the nonlinear control system problems, CNFN-ISEL is adopted to design controllers in four simulations - multi-input single-output plant control, multi-input multi-output plant control [9] and backing up the truck [10]. Table I presents the initial parameters before training used in the four computer simulations.

Example 1: Multi-input Single Output (MISO) Control

The controlled plant is the same as that used elsewhere [9], and is described by

$$y_p(k+1) = f[y_p, y(k-1)] + u(k) \quad (23)$$

$$\text{where } f[y_p, y(k-1)] = \frac{y_p(k)y_p(k-1)[y_p(k)+2.5]}{1+y_p^2(k)+y_p^2(k-1)}.$$

In designing the CNFN, the desired output y_r is specified by the following 250 pieces of data;

$$y_r(k+1) = 0.6y_r(k) + 0.2y_r(k-1) + r(k) \quad (24)$$

$$\text{where } r(k) = 0.5\sin(2\pi k/45) + 0.2\sin(2\pi k/15) + 0.2\sin(2\pi k/90).$$

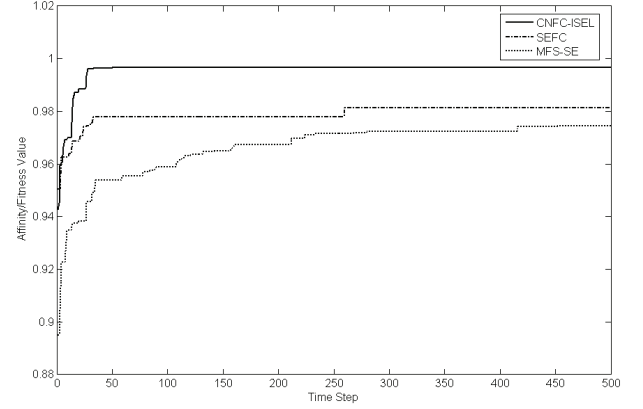


Figure 10. Learning curves of best performance of the CNFN-ISEL, SEFC and MFS-SE in Example 1.

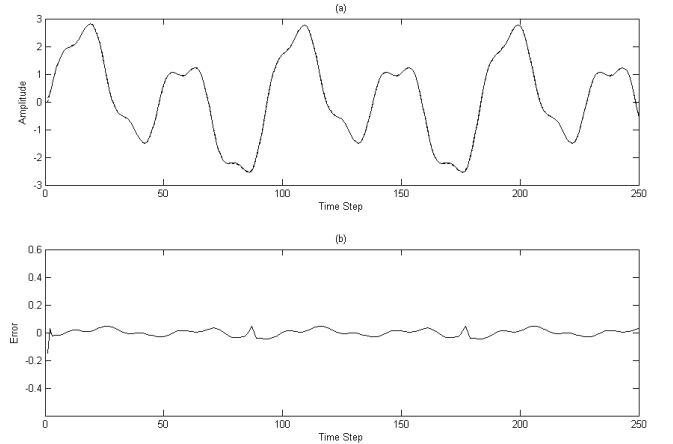


Figure 11. (a) Desired (solid line) and model (dotted line) output generated by CNFN designed with ISEL in Example 1. (b) Errors of proposed CNFN-ISEL.

The inputs to the CNFN are $y_p(k)$, $y_p(k+1)$, $y_r(k)$ and $y_r(k+1)$ and the output is $u(k)$. For the self-clustering algorithm (SCA), we chose the parameter $D_{thr}=0.85$ and then four rules are generated by SCA in CNFN; hence, $R=4$, and the number of total parameters is 60. Accordingly, the number of genes is also 15 in each subgroup. Figure 10 plots the learning curves of the best performance of the CNFN-ISEL for the affinity/fitness value, the symbiotic evolution of the fuzzy controller (SEFC) [7] and the Mamdani-type fuzzy system using the symbiotic evolution algorithm (MFS-SE) [8], after the learning process of 500 generations. Figure 11 (a) plots the control results of the desired output and the model output after a learning process of 500 generations. Figure 11 (b) shows the errors of the proposed method. Table II presents the best and averaged affinity/fitness

values after 500 generations of training. The comparison indicates that the best and averaged affinity/fitness values of CNFN-ISEL are better than those obtained using other methods.

TABLE II. PERFORMANCE COMPARISONS WITH CNFN-ISEL, SEFC AND MFS-SE IN EXAMPLE 1.

Method	CNFN-ISEL	SEFC [7]	MFS-SE [8]
Affinity/Fitness Value (Ave)	0.9916	0.9793	0.9655
Affinity/Fitness Value (Best)	0.9961	0.9802	0.9745

Example 2: Multi-input Multi-output (MIMO) Control

In this example, the MIMO plants [9] to be controlled are described by the equations

$$\begin{bmatrix} y_{p1}(k+1) \\ y_{p2}(k+1) \end{bmatrix} = \begin{bmatrix} 0.5 \frac{y_{p1}(k)}{1 + y_{p2}^2(k)} \\ 0.5 \frac{y_{p1}(k)y_{p2}(k)}{1 + y_{p2}^2(k)} \end{bmatrix} + \begin{bmatrix} u_1(k) \\ u_2(k) \end{bmatrix}. \quad (25)$$

The controlled outputs should follow the desired output y_{r1} and y_{r2} as specified by the following 250 pieces of data;

$$\begin{bmatrix} y_{r1}(k) \\ y_{r2}(k) \end{bmatrix} = \begin{bmatrix} \sin(k\pi/45) \\ \cos(k\pi/45) \end{bmatrix}. \quad (26)$$

The inputs of the CNFN are $y_{p1}(k)$, $y_{p2}(k)$, $y_{r1}(k)$ and $y_{r2}(k)$, and the outputs are $u_1(k)$ and $u_2(k)$. For the self-clustering algorithm (SCA), we chose the parameter $D_{thr}=1$ and then six rules generated by SCA in CNFN, $R=6$. Therefore, the total number of parameters is 120 and the number of genes is 20 in each subgroup.

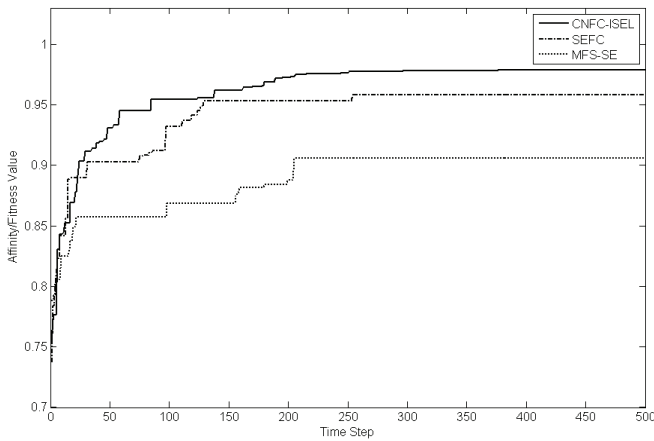


Figure 12. Learning curves of best performance of the CNFN-ISEL, SEFC and MFS-SE in Example 2.

Figure 12 plots the learning curves of the best performance of the CNFN-ISEL model for the affinity/fitness value, the symbiotic evolution of the fuzzy controller (SEFC) [7] and the Mamdani-type fuzzy system using symbiotic evolution algorithm (MFS-SE) [8], after the learning process of 500 generations. To demonstrate the control result, Figures 13(a)

and (b) plot the control results of the desired output and the model output after the learning process of 500 generations. Figures 13 (c) and (d) show the errors of the proposed method. Table III presents the best and averaged affinity/fitness values after 500 generations of training. The comparison indicates that the best and averaged affinity/fitness values of CNFN-ISEL are better than those of other methods.

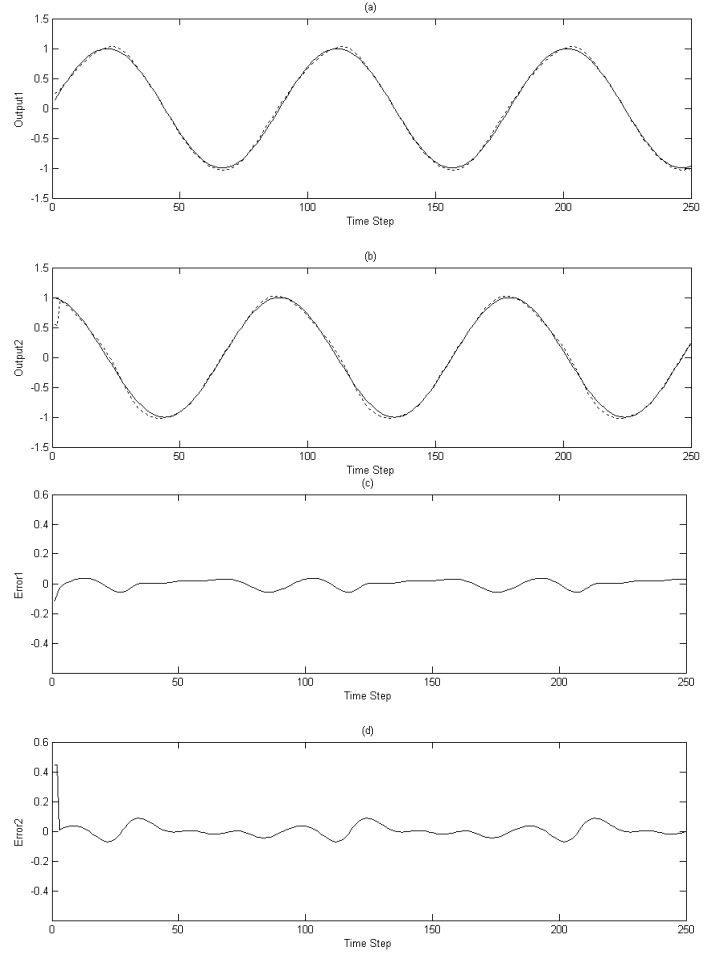


Figure 13. Desired (solid line) and model (dotted line) output generated by CNFN designed with ISEL in Example 1. (a) Output 1. (b) Output 2. Errors of proposed CNFN-ISEL for (c) output 1 and (d) output 2.

TABLE III. COMPARISON OF PERFORMANCE OF CNFN-ISEL, SEFC AND MFS-SE IN EXAMPLE 2.

Method	CNFN-ISEL	SEFC [7]	MFS-SE [8]
Affinity/Fitness Value (Ave)	0.9721	0.9553	0.8503
Affinity/Fitness Value (Best)	0.9786	0.9581	0.8560

Example 3: Control of Backing Up the Truck

Backing a truck into a loading dock is difficult. It is a nonlinear control problem for which no traditional control method exists [10]. Figure 14 shows the simulated truck and loading zone. The truck position is exactly determined by three state variables ϕ , x and y , where ϕ is the angle between the truck and the horizontal, and the coordinate pair (x, y) specifies

the position of the center of the rear of the truck in the plane. The steering angle θ of the truck is the controlled variable. Positive values of θ represent clockwise rotations of the steering wheel and negative values represent counterclockwise rotations. The truck is placed at some initial position and is backed up while being steered by the controller. The objective of this control problem is to use backward only motions of the truck to make the truck arrive in the desired loading dock $(x_{\text{desired}}, y_{\text{desired}})$ at a right angle ($\phi_{\text{desired}}=90^\circ$). The truck moves backward as the steering wheel moves through a fixed distance (df) in each step. The loading region is limited to the plane $[0,100] \times [0,100]$.

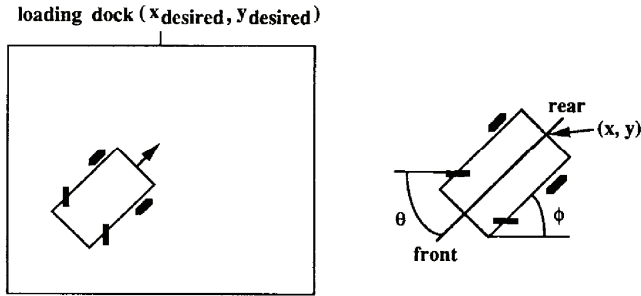


Figure 14. Diagram of simulated truck and loading zone.

The input and output variables of the CNFN must be specified. The controller has two inputs, truck angle ϕ and cross position x . When the clearance between the truck and the loading dock is assumed to be sufficient, the y coordinate is not considered as an input variable. The output of the controller is the steering angle θ . The ranges of the variables x , ϕ and θ are as follows.

$$0 \leq x \leq 100 \quad (27)$$

$$-90^\circ \leq \phi \leq 270^\circ \quad (28)$$

$$-30^\circ \leq \theta \leq 30^\circ \quad (29)$$

The equations of backward motion of the truck are,

$$\begin{aligned} x(k+1) &= x(k) + d_f \cos \theta(k) + \cos \phi(k) \\ y(k+1) &= y(k) + d_f \cos \theta(k) + \sin \phi(k) \end{aligned} \quad (30)$$

$$\phi(k+1) = \tan^{-1} \left[\frac{l \sin \phi(k) + d_f \cos \phi(k) \sin \theta(k)}{l \cos \phi(k) - d_f \sin \phi(k) \sin \theta(k)} \right]$$

where l is the length of the truck. Equation (30) yields the next state from the present state.

Learning involves several attempts, each starting from an initial state and terminating when the desired state is reached; the CNFN is thus trained. For the self-clustering algorithm (SCA), we chose the parameter $D_{thr}=0.45$. After SCA learning, 13 fuzzy logic rules are generated and the parameters of the membership functions are determined. Hence, the total number of parameters is 117 and the number of genes is 9 in each subgroup. The training process continues for 500 generations. The affinity of the CNFN is approximately 0.95587 and the learning curve of CNFN is compared with those obtained using various existing models [7], [8], as shown in Fig. 15. Figures 16(a)-(b) plot the trajectories of the moving truck controlled by

the CNFN, starting at initial positions $(x, y, \phi) = (a) (10, 20, 150^\circ)$ and $(b) (70, 20, -30^\circ)$, after the training process has been terminated. The considered performance indices include the best affinity/fitness and the average affinity/fitness value. Table IV compares the results. According to these results, the proposed CNFN-ISEL outperforms various existing models.

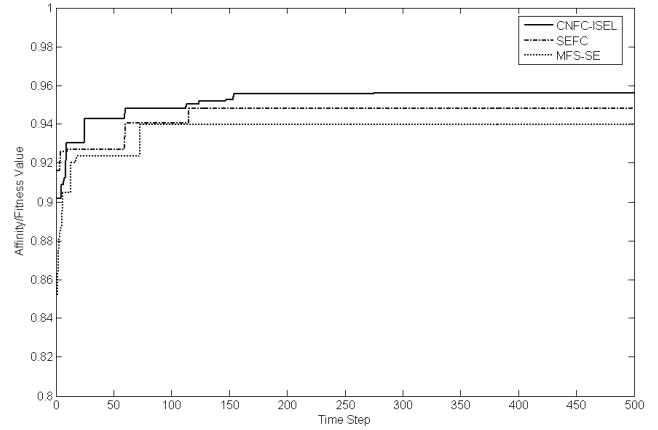


Figure 15. Learning curves of best performance of the CNFN-ISEL, SEFC and MFS-SE in Example 3.

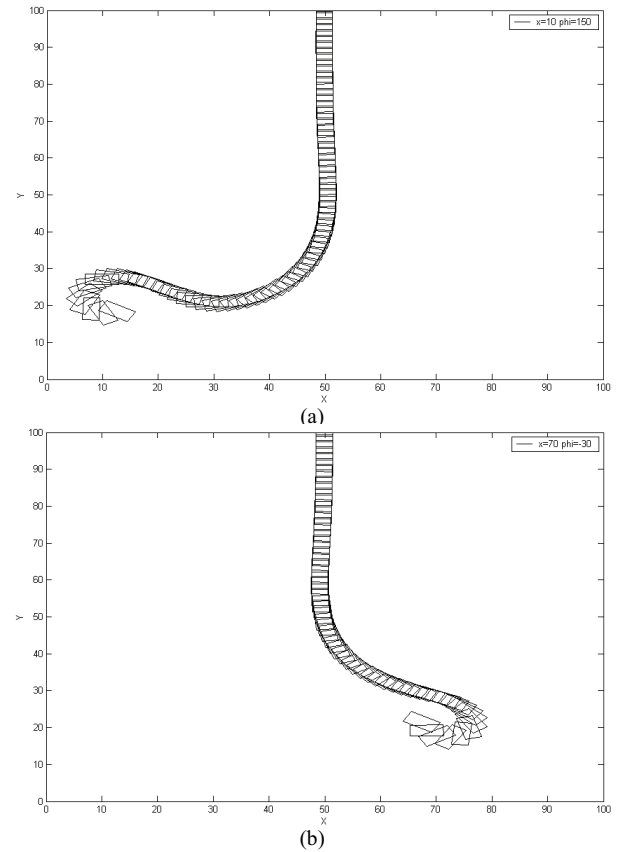


Figure 16. Trajectories of truck, starting at four initial positions under the control of the CNFN-ISEL after learning using training trajectories.

TABLE IV. COMPARISONS OF PERFORMANCE OF CNFN-ISEL, SEFC AND MFS-SE IN EXAMPLE 3.

Method	CNFN-ISEL	SEFC [7]	MFS-SE [8]
Affinity/Fitness Value (Ave)	0.9511	0.9451	0.9332
Affinity/Fitness Value (Best)	0.9558	0.9516	0.9398

V. CONCLUSION

This study proposes an efficient immune symbiotic evolution learning algorithm for the compensatory neural fuzzy network. The immune symbiotic evolution learning method (ISEL) includes three major components - initial population, subgroup symbiotic evolution and immune system algorithm. However, the self-clustering algorithm, an evolution of the rule itself, the clonal selection principle of antibodies, and self-adaptive mutation are achieved by ISEL method. Moreover, the proposed CNFN-ISEL obtains better simulation results than alternative models in some circumstances, for example achieving faster learning and higher design accuracy in many nonlinear control problems.

ACKNOWLEDGMENT

This research was sponsored by the National Science Council, Taiwan, R.O.C., under Grant NSC 99-2218-E-150-047.

REFERENCES

- [1] Y. Shi, R. Eberhart and Y. Chen, "Implementation of evolutionary fuzzy systems," *IEEE Trans. Fuzzy Syst.*, vol. 7, pp. 109-119, Apr. 1999.
- [2] T. L. Seng, M. B. Khalid, and R. Yusof, "Tuning of a neuro-fuzzy controller by genetic algorithm," *IEEE Trans. Syst., Man, Cybern. B*, vol. 29, pp. 226-236, Apr. 1999.
- [3] H. K. Lam, F. H. F. Leung, and P. K. S. Tam, "Design and stability analysis of fuzzy model based nonlinear controller for nonlinear systems using genetic algorithm," *IEEE Trans. Syst., Man, Cybern. B*, vol. 33, pp. 250-257, Apr. 2003.
- [4] C. J. Lin and C. H. Chen, "Identification and prediction using recurrent compensatory neuro-fuzzy systems," *Fuzzy Sets and Systems*, vol. 150, no. 2, pp.307-330, Mar. 2005.
- [5] C. J. Lin, C. H. Chen and C. Y. Lee, "A self-adaptive quantum radial basis function network for classification applications," *Proc. IEEE International Joint Conference Neural Networks*, vol. 4, pp. 25-29, 2004.
- [6] O. Cordon, F. Herrera, F. Hoffmann, and L. Magdalena, "Advances in Fuzzy Systems-Applications and Theory," *Genetic fuzzy systems evolutionary tuning and learning of fuzzy knowledge bases.*, vol. 19, NJ: World Scientific Publishing, 2001.
- [7] C. F. Juang, J. Y. Lin and C. T. Lin, "Genetic reinforcement learning through symbiotic evolution for fuzzy controller design," *IEEE Trans. Syst., Man, Cybern. B*, vol. 30, no. 2, Apr. 2000.
- [8] M. Jamei, M. Mahfouf and D. A. Linkens, "Elicitation and fine-tuning of fuzzy control rules using symbiotic evolution," *Fuzzy Sets and Systems*, vol. 147, no. 1, pp. 57-74, Oct. 2004.
- [9] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Trans. Neural Networks*, vol. 1, no. 1, pp. 4-27, Mar. 1990.
- [10] D. Nguyen and B. Widrow, "The truck backer-upper: An example of self-learning in neural network," *IEEE Conf. Syst. Mag.*, vol. 10, no. 3, 1990, pp. 18-23.