International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:9, No:4, 2015

Evolution of Fuzzy Neural Networks Using an Evolution Strategy with Fuzzy Genotype Values

Hidehiko Okada

Abstract—Evolution strategy (ES) is a well-known instance of evolutionary algorithms, and there have been many studies on ES. In this paper, the author proposes an extended ES for solving fuzzy-valued optimization problems. In the proposed ES, genotype values are not real numbers but fuzzy numbers. Evolutionary processes in the ES are extended so that it can handle genotype instances with fuzzy numbers. In this study, the proposed method is experimentally applied to the evolution of neural networks with fuzzy weights and biases. Results reveal that fuzzy neural networks evolved using the proposed ES with fuzzy genotype values can model hidden target fuzzy functions even though no training data are explicitly provided. Next, the proposed method is evaluated in terms of variations in specifying fuzzy numbers as genotype values. One of the mostly adopted fuzzy numbers is a symmetric triangular one that can be specified by its lower and upper bounds (LU) or its center and width (CW). Experimental results revealed that the LU model contributed better to the fuzzy ES than the CW model, which indicates that the LU model should be adopted in future applications of the proposed method.

Keywords—Evolutionary algorithm, evolution strategy, fuzzy number, feedforward neural network, neuroevolution.

I. Introduction

A multi-layered feedforward neural network (NN) with fuzzy-valued weights and biases has been proposed in the literature [1]. The fuzzy NN (FNN) approximately models a fuzzy function $Y = F(\boldsymbol{x})$, where Y is a fuzzy number and $\boldsymbol{x}$ is a real vector, by learning the given data $(\boldsymbol{x}_1, Y_1), (\boldsymbol{x}_2, Y_2), \ldots$. The FNN can learn the data in which $Y_1, Y_2, \ldots$ include both real numbers and fuzzy numbers, because a real number can be specified as a fuzzy number with zero width (i.e., with the same value of the upper and lower bounds). As the learning method for the FNNs, a supervised learning method has also been proposed [1], which is an extension of the traditional back propagation (BP) algorithm, but a method that does not require training data has not been proposed as yet.

Besides, evolutionary algorithms have recently been applied to the reinforcement training of NNs, known as neuroevolution (NE) [2]-[5]. In NE, weights and biases are tuned by evolutionary operations, not by the BP algorithm. Because NE does not utilize BP, NE does not have errors between NN output values and their target signals but requires each NN to be ranked on the basis of its performance for a given task. Thus, NE is applicable to problems in which the error function is difficult or impossible to determine, such as controlling autonomous robots. EAs have been applied to the NE of traditional NNs with real-valued weights and biases, where the genotypes (chromosomes) consist of real numbers or bit strings that encode real numbers. Ordinary EAs do not employ fuzzy numbers as their genotype values because their evolutionary operations are designed to handle genotypes with crisp values, and thus, the operations cannot handle genotypes with fuzzy values.

Hidehiko Okada is with Faculty of Computer Science and Engineering, Kyoto Sangyo University, Kamigamo Motoyama, Kita-ku, Kyoto 603-8555, Japan (e-mail: hidehiko@cc.kyoto-su.ac.jp).

The author previously proposed an extension of the genetic algorithm (GA) that can handle fuzzy-valued genotypes [6]. In this paper, the author proposes a similar extension of another EA, the evolution strategy (ES). ES [7][8] is a well-known instance of the EAs, and there have been many studies on ES [9]-[25]. For example, Hansen et al. proposed an ES variant, CMA-ES, which adopts mutation based on normal distributions [9][16]. The extended ES proposed in this paper can be applied directly to fuzzy optimization problems by employing fuzzy variables as genotype values. The author experimentally applies the proposed method (fuzzy-valued ES: FES) to the reinforcement training of FNNs and compares the experimental result with the result of the previously proposed fuzzy-valued GA [6].

II. Neural Networks with Fuzzy Weights and Biases

The FNN employed in this research is the same as in the literature [1], which is a three-layered feedforward NN with fuzzy weights and biases. Fig. 1 shows its structure. An FNN receives an input real vector $\boldsymbol{x}$ and calculates its output fuzzy value O (for simplicity, the output layer includes a single unit) as follows [1]:

Input layer:

$$o_i = x_i. \tag{1}$$

Hidden layer:

$$Net_j = \sum_i W_{j,i} o_i + \Theta_j, \tag{2}$$

$$O_j = f(Net_j). \tag{3}$$

Output layer:

$$Net = \sum_j W_j O_j + \Theta, \tag{4}$$

$$O = f(Net). \tag{5}$$

In (1)-(5), x_i and o_i denote real values, while Net_j, Net, $W_{j,i}$, W_j, Θ_j, Θ, O_j, and O represent fuzzy values. $f(x)$ denotes the unit activation function, which is typically the sigmoidal one: $f(x) = 1/(1+e^{-x})$. $f(x)$ maps a fuzzy input number to a fuzzy output number, as illustrated in Fig. 2.

The feedforward calculation of the FNN is based on the extension principle [26] and the interval arithmetic [27] (for

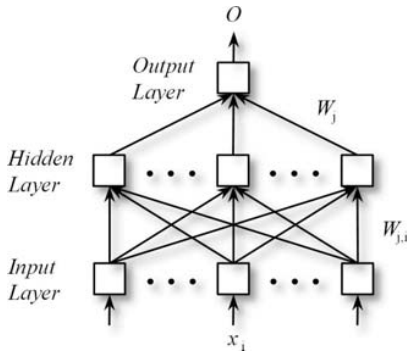

Fig. 1: Neural network with fuzzy weights and biases [1].

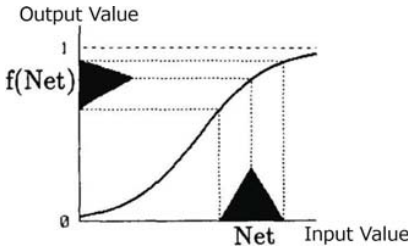

Fig. 2: Input-output relation of each unit in the hidden and output layers [1].

more details, see the literature [1]). Let us denote two closed intervals as A and B, where $A = [a^L, a^U]$ and $B = [b^L, b^U]$. In this case,

$$\begin{aligned} A+B &= [a^L, a^U] + [b^L, b^U] \\ &= [a^L + b^L, a^U + b^U]. \end{aligned} \quad (6)$$

$$\begin{aligned} k \cdot A &= k \cdot [a^L, a^U] \\ &= [ka^L, ka^U], \text{if } k \geq 0, \text{else } [ka^U, ka^L]. \end{aligned} \quad (7)$$

$$\begin{aligned} A \cdot B &= [a^L, a^U] \cdot [b^L, b^U] \\ &= [min(a^L b^L, a^L b^U, a^U b^L, a^U b^U), \\ &\quad max(a^L b^L, a^L b^U, a^U b^L, a^U b^U)]. \end{aligned} \quad (8)$$

III. Proposed Method: Evolution Strategy with Fuzzy Genotype Values

Suppose that the FNN shown in Fig. 1 includes n input units and m hidden units. In this case, the FNN includes $mn+m$ weights (i.e., mn weights between the input and hidden layers, and m weights between the hidden and output layers) and $m+1$ biases (i.e., $m+1$ is the total number of units in the hidden and output layers). Thus, the FNN includes $mn+2m+1$ fuzzy variables in total. The FES proposed in this paper handles these fuzzy variables as a genotype $\boldsymbol{X} = (X_1, X_2, ..., X_D)$, where X_i denotes a fuzzy number and $D = mn+2m+1$. Suppose that each X_i represents a symmetric triangular fuzzy number (Fig. 3) as in [6]. In this case, X_i can be specified by its upper and lower bounds or by its center and width (radius): $X_i = [x_i^L, x_i^U]$ or $X_i = (x_i^c, x_i^w)$, where x_i^L, x_i^U, x_i^c, and x_i^w denote the upper, lower, center, and width of X_i, respectively.

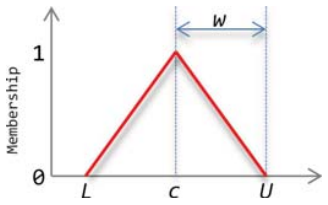

Fig. 3: Symmetric triangular fuzzy number and its real-valued parameters [6].

The FES includes the same processes as those in the ordinary ES with real-valued genotypes. Processes of the initialization of population, reproduction, and fitness evaluation are extended so that these processes can handle fuzzy-valued genotypes.

A. Initialization of Population

In the initialization process, $\boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_P$ are randomly initialized, where P denotes the population size. Because the elements in $\boldsymbol{X}_a$ (i.e., $X_{a,1}, X_{a,2}, ..., X_{a,D}$) are weights and biases in an FNN in this research, smaller absolute values of $X_{a,i}$ are preferable as the initial values. Thus, the initial values for $X_{a,i}$ are randomly sampled from the normal distribution $N(0, \varepsilon_1)$ or uniformly sampled from an interval $[-\varepsilon_1, \varepsilon_1]$, where ε_1 denotes a small positive number. In the case of employing the [lower, upper] model (the LU model), two values are sampled per $X_{a,i} = [x_{a,i}^L, x_{a,i}^U]$: the smaller (larger) one is set to $x_{a,i}^L$ ($x_{a,i}^U$). In the case of employing the (center, width) model (the CW model), two values are sampled per $X_{a,i} = (x_{a,i}^c, x_{a,i}^w)$: one of the two values is set to $x_{a,i}^c$, and the absolute value of the other is set to $x_{a,i}^w$.

B. Reproduction

Let us denote a parent genotype instance as $\boldsymbol{X}_a$ and an offspring genotype instance as $\boldsymbol{X}_z$. $\boldsymbol{X}_a$ can be sampled from the population in the same manner as the ordinary ES.

In the case of employing the LU model,

$$\boldsymbol{X}_a = (X_{a,1}, X_{a,2}, ..., X_{a,D}), \quad (9)$$

$$X_{a,i} = [x_{a,i}^L, x_{a,i}^U], \quad (10)$$

$$\boldsymbol{X}_z = (X_{z,1}, X_{z,2}, ..., X_{z,D}), \quad (11)$$

$$X_{z,i} = [x_{z,i}^L, x_{z,i}^U]. \quad (12)$$

$x_{z,i}^L$ and $x_{z,i}^U$ are determined as follows:

$$x_{z,i}^L = x_{a,i}^L + rand, \quad (13)$$

$$x_{z,i}^U = x_{a,i}^U + rand, \quad (14)$$

where $rand$ denotes a random number. The value for $rand$ is sampled from the normal distribution $N(0, \varepsilon_2)$ or uniformly

sampled from an interval $[-\varepsilon_2, \varepsilon_2]$ every time when *rand* is called. ε_2 denotes another small positive number.

In the case of employing the CW model,

$$\boldsymbol{X}_a = (X_{a,1}, X_{a,2}, ..., X_{a,D}), \tag{15}$$

$$X_{a,i} = (x^c_{a,i}, x^w_{a,i}), \tag{16}$$

$$\boldsymbol{X}_z = (X_{z,1}, X_{z,2}, ..., X_{z,D}), \tag{17}$$

$$X_{z,i} = (x^c_{z,i}, x^w_{z,i}). \tag{18}$$

$x^c_{z,i}$ and $x^w_{z,i}$ are determined as follows:

$$x^c_{z,i} = x^c_{a,i} + rand, \tag{19}$$

$$x^w_{z,i} = x^w_{a,i} + rand, \tag{20}$$

where *rand* denotes the same random number as that for the LU model.

Note that $x^L_{z,i}$ must not be larger than $x^U_{z,i}$ because $x^L_{z,i}$ and $x^U_{z,i}$ denote the lower and upper bounds of the fuzzy number $X_{z,i}$, respectively. Similarly, $x^w_{z,i}$ must not be negative because $x^w_{z,i}$ denotes the width of $X_{z,i}$. If the value of $x^L_{z,i}$ becomes larger than the value of $x^U_{z,i}$ obtained using (13) and (14), these values must be repaired to meet the constraint. The repair method can be described as follows:

- the value of $x^U_{z,i}$ is assigned to $x^L_{z,i}$,
- the value of $x^L_{z,i}$ is assigned to $x^U_{z,i}$,
- the mean value of $x^L_{z,i}$ and $x^U_{z,i}$ is calculated and assigned to both of $x^L_{z,i}$ and $x^U_{z,i}$, respectively, or
- the two values for $x^L_{z,i}$ and $x^U_{z,i}$ are interchanged.

Similarly, if the value of $x^w_{z,i}$ becomes negative by the use of (20), the value must be repaired to meet the constraint. The repair method can be described as follows:

- the value of $x^w_{z,i}$ is set to 0, or
- the absolute value of $x^w_{z,i}$ is assigned to $x^w_{z,i}$.

C. Fitness Evaluation

To evaluate the fitness of an FNN as a phenotype instance of the corresponding genotype instance $\boldsymbol{X}_j = (X_{j,1}, X_{j,2}, ..., X_{j,D})$, where $\boldsymbol{X}_j \in \{\boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_P\}$, the FNN is supplied with several samples of input real vectors and calculates output values. The input values are sampled within the variable domain of the application problem. The fitness of the genotype instance $\boldsymbol{X}_j$ is evaluated on the basis of the output values. The method for scoring the fitness based on the output values depends on the problem to which the FNN is applied. For example, in a case where the FNN is applied to control an automated system, one of the performance measures of the system can be used as the fitness score of the genotype instance corresponding to the FNN.

IV. Application to Evolving Fuzzy Neural Networks

The author experimentally evaluates the ability of the FES by applying it to the evolution of FNNs in the same manner as in [6]. The FNNs are challenged to model hidden fuzzy functions. The author adopts the same three functions [6] as the targets for FNNs to model so that the author can compare the experimental result with that obtained using the fuzzy GA (FGA) [6].

In Section IV.A, the modelling accuracy is evaluated by using three target functions. In Section IV.B, the LU model is compared with the CW model to investigate which model better contributes to the FES in evolving FNNs. In Section IV.C, the FES is compared with the FGA.

A. Accuracy of Modeling Fuzzy Functions

The author employs the same three fuzzy functions [6] as the targets for FNNs to model. For simplicity, the input x of the target functions is not a real vector but a real scalar (therefore, the FNN includes only a single input unit) and $0 \le x \le 1$, as in literature [1]. The outputs of the target functions are symmetric triangular fuzzy numbers. The functions $F_1(x) = [F_1(x)^L, F_1(x)^U]$, $F_2(x) = [F_2(x)^L, F_2(x)^U]$, and $F_3(x) = [F_3(x)^L, F_3(x)^U]$ can be expressed as follows:

$$F_1(x)^L = 0.2sin(2\pi x) - 0.1x^2 + 0.4, \tag{21}$$

$$F_1(x)^U = 0.2sin(2\pi x) + 0.1x^2 + 0.6. \tag{22}$$

$$F_2(x)^L = 0.2sin(2\pi x) + 0.2x^2 + 0.2, \tag{23}$$

$$F_2(x)^U = 0.2sin(2\pi x) - 0.2x^2 + 0.7. \tag{24}$$

$$F_3(x)^L = 0.15sin(2\pi x) + 0.3, \tag{25}$$

$$F_3(x)^U = 0.1sin(3\pi x) - 0.1x + 0.7. \tag{26}$$

Figs. 4-6 show these three functions, where:

- F0.0L and F0.0U denote $F(x)^L$ and $F(x)^U$, i.e., the lower and upper bounds of the support interval of $F(x)$, respectively;
- F0.5L and F0.5U denote the lower and upper bounds, respectively, of the 0.5-level interval of $F(x)$, i.e., $F(x)|_{0.5}$; and
- F1.0 denotes the peak of $F(x)$, i.e., $F(x)|_{1.0}$.

The FNN is designed as follows [6]:

- Number of units: 1 input, 10 hidden, 1 output
- $-10.0 \le x^L_{j,i}, x^U_{j,i}, x^c_{j,i} \le 10.0$
- $0.0 \le x^w_{j,i} \le 10.0$

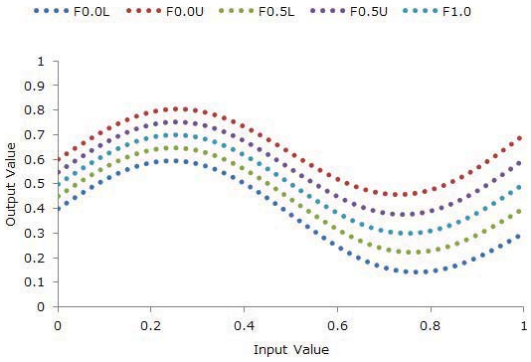

Fig. 4: Target function $F_1(x)$ [6].

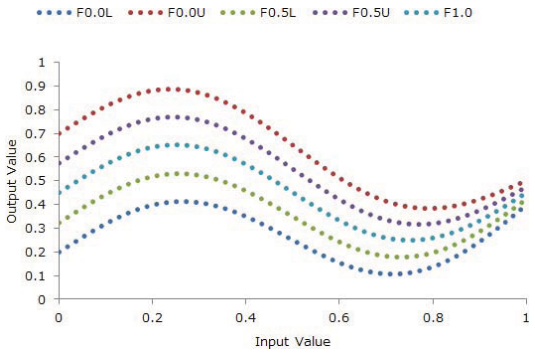

Fig. 5: Target function $F_2(x)$ [6].

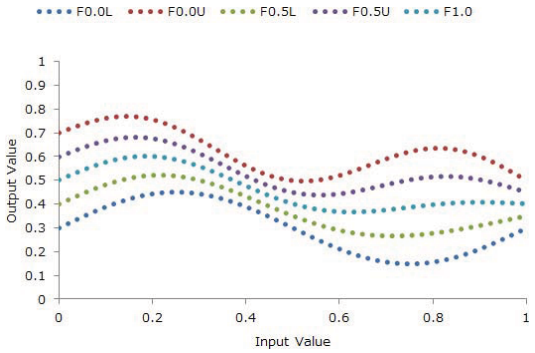

Fig. 6: Target function $F_3(x)$ [6].

The FES is designed as follows:

- Total number of FNNs evolved in a single run: 1,000,000
- Population size and the total number of generations: (10,100)-ES with 10,000 generations, or (50,500)-ES with 2,000 generations
- Initial values of $x^L_{j,i}$, $x^U_{j,i}$, and $x^c_{j,i}$ for the fuzzy weights and biases: uniformly random within $[-1.0, 1.0]$, i.e., $\varepsilon_1 = 1.0$.
- Initial values for $x^w_{j,i}$: uniformly random within $[0.0, 1.0]$.
- Random values for the reproduction: $N(0, 0.1)$, i.e., $\varepsilon_2 = 0.1$.

Genotype instances $\boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_P$ are ranked by utilizing the same error function as that in literature [1][6]. As the values for the h-level intervals of fuzzy numbers, the author employs $h = 0.2, 0.4, ..., 1.0$ in this experiment. A phenotype instance FNN that corresponds to a genotype instance $\boldsymbol{X}_j$ is supplied with a real input value x_r and calculates its output fuzzy number O_r. x_r is sampled within the input domain $[0, 1]$ as $x_r = 0.0, 0.01, 0.02, ..., 1.0$. Besides, each value of x_r is supplied to the target function $F(x)$ and the output fuzzy number $F(x_r)$ is obtained. Then, the cost e_r for the input x_r is calculated as follows:

$$e_r = \sum_h h((o^L_{r,h} - f^L_{r,h})^2 + (o^U_{r,h} - f^U_{r,h})^2), \tag{27}$$

where

- $o^L_{r,h}$ and $o^U_{r,h}$ denote the lower and upper bounds, respectively, of the h-level interval of O_r, i.e., $O_r|_h = [o^L_{r,h}, o^U_{r,h}]$, and
- $f^L_{r,h}$ and $f^U_{r,h}$ represent the lower and upper bounds, respectively, of the h-level interval of $F(x_r)$, i.e., $F(x_r)|_h = [f^L_{r,h}, f^U_{r,h}]$.

For each genotype instance $\boldsymbol{X}_j$, e_r is calculated 101 times $(e_0, e_1, ..., e_{100})$ for the 101 input values $x_r = 0.0, 0.01, 0.02, ..., 1.0$, and the sum of the calculated e_r values is used for ranking $\boldsymbol{X}_j$. An instance with a smaller sum of e_r is ranked higher. Note that e_r scores are utilized not for reproducing new FNNs (i.e., updating FNN weights and biases) but for ranking FNNs in the current population.

Figs. 7-9 show the results of this experiment. Fig. 7 shows the output fuzzy function of the best FNN among the total 20,000,000 FNNs (= [1,000,000 FNNs in each run] $*$ [five runs] $*$ [two variations for population sizes] $*$ [two variations for the interval model]) evolved by the FES for modeling $F_1(x)$. Figs. 8 and 9 show those for modeling $F_2(x)$ and $F_3(x)$, respectively, in the same manner as Fig. 7. In Figs. 7-9,

- F0.0L, F0.0U, F0.5L, F0.5U, and F1.0 are the same as those in Figs. 4-6,
- NN0.0L and NN0.0U denote the lower and upper bounds, respectively, of the support interval of the FNN output fuzzy number,
- NN0.5L and NN0.5U denote the lower and upper bounds, respectively, of the 0.5-level interval of the FNN output fuzzy number, and
- NN1.0 denotes the peak of the FNN output fuzzy number.

Fig. 10 shows the membership functions of O and $F_1(x)$ for the input values $x = 0.2$ and $x = 0.8$, where O denotes the output fuzzy number of the best FNN. In this figure,

- NN(0.2) and NN(0.8) represent the membership functions of the output fuzzy number of the best FNN for the input values of 0.2 and 0.8, while
- F(0.2) and F(0.8) denote the membership functions of $F_1(x)$ for the input values of 0.2 and 0.8.

Figs. 11 and 12 show those for $F_2(x)$ and $F_3(x)$ in the same manner as Fig. 10. The shapes of the FNN output fuzzy numbers (the solid curves in Figs. 10-12) are similar to those of the target fuzzy numbers (the dotted lines in the same figures) for larger values of the membership score. These results are shown in Figs. 7-12 and reveal that the best

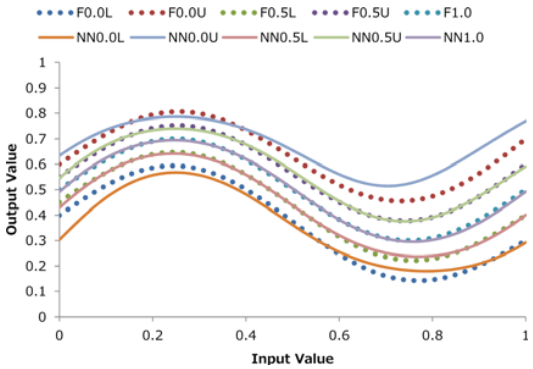

Fig. 7: Output fuzzy function of the best FNN evolved by FES for modeling $F_1(x)$.

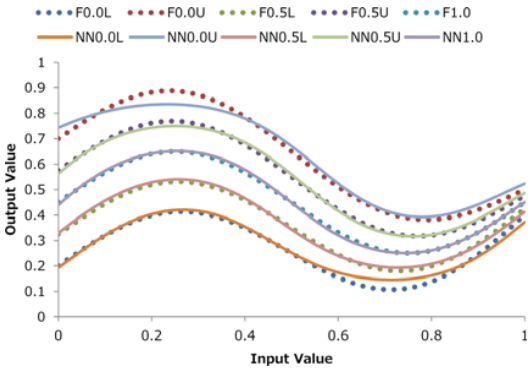

Fig. 8: Output fuzzy function of the best FNN evolved by FES for modeling $F_2(x)$.

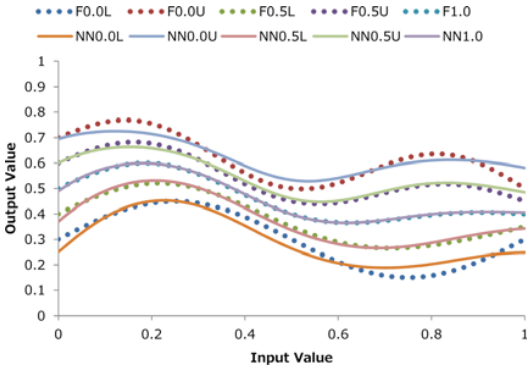

Fig. 9: Output fuzzy function of the best FNN evolved by FES for modeling $F_3(x)$.

FNNs evolved by the FES approximate their target functions (particularly for larger membership scores because the error is weighted more for the larger membership scores, see (27)), despite the fact that no training data are explicitly provided.

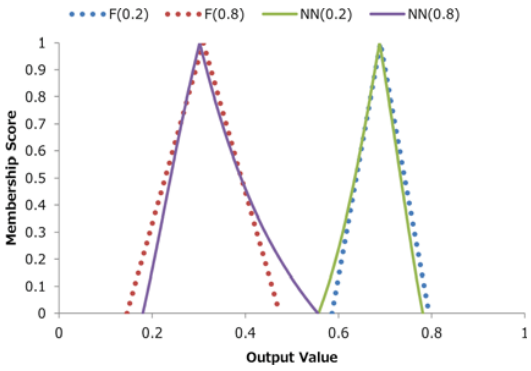

Fig. 10: Output fuzzy numbers of the best FNN evolved by FES and target fuzzy numbers $F_1(x)$ for the inputs values of 0.2 and 0.8.

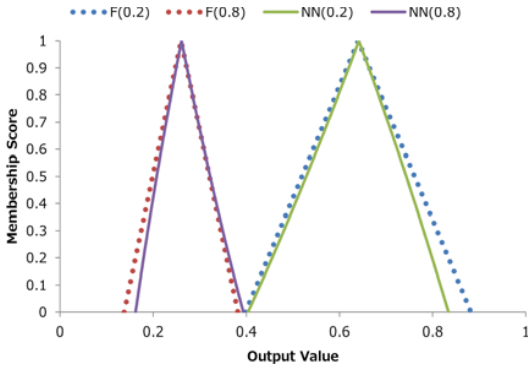

Fig. 11: Output fuzzy numbers of the best FNN evolved by FES and target fuzzy numbers $F_2(x)$ for the inputs values of 0.2 and 0.8.

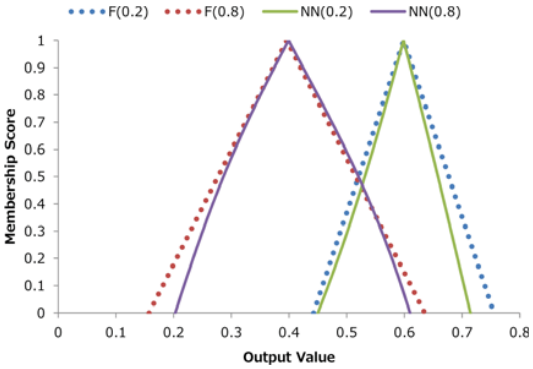

Fig. 12: Output fuzzy numbers of the best FNN evolved by FES and target fuzzy numbers $F_3(x)$ for the inputs values of 0.2 and 0.8.

B. Comparison of Two Models for Fuzzy Genotype Values

As described in Section III.B, the constraints for the two real parameters of a symmetric triangular fuzzy number (i.e., the lower and upper bounds or the center and the width) are different: note that the width must not be negative, while the other three parameters can be negative. Because of the difference in the constraints, the search space for the FES with the LU model is not the same as that with the CW model even for the same task. The difference in the search spaces between the two models may affect the performance of the FES in searching for solutions: a smaller search space is usually more advantageous for any evolutionary algorithm. In this section, the author compares the two models to investigate which model contributes better to the FES with respect to finding better solutions, on the basis of the result of the numerical experiments described in the previous section.

Fig. 13 shows the error value of the FNN that is the best among each total number of FNNs evolved for $F_1(x)$ (e.g., 500,000 FNNs are evolved in total at the 5,000th generation by (10,100)-ES). In this figure, "LU (100)" denotes the result obtained by using (10,100)-ES with the LU model. "LU (500)", "CW (100)", and "CW (500)" denote results in the

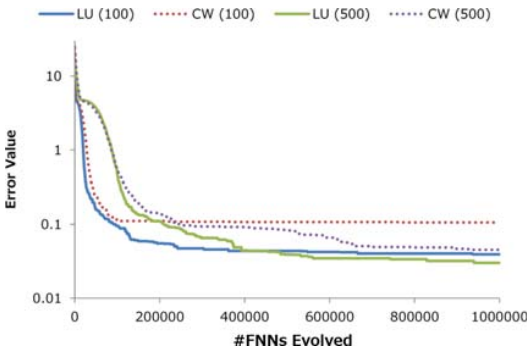

Fig. 13: Error value of the best FNN for each total number of FNNs evolved for modeling $F_1(x)$.

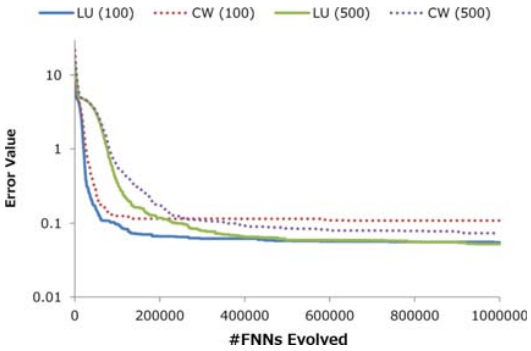

Fig. 14: Error value of the best FNN for each total number of FNNs evolved for modeling $F_2(x)$.

same manner as "LU (100)." The error values are the averaged values of five runs. Figs. 14 and 15 show the error values for $F_2(x)$ and $F_3(x)$, respectively, in the same manner as Fig. 13.

Figs. 13-15 reveal that for all three target functions, the LU model contributed better than the CW model in the cases of both (10,100)-ES and (50,500)-ES, i.e., after the evolution of 1,000,000 FNNs, the solid curves for the LU model went below the dotted curves for the CW model. This finding suggests that the LU model is better for the FES to employ as the model for specifying symmetric triangular fuzzy numbers

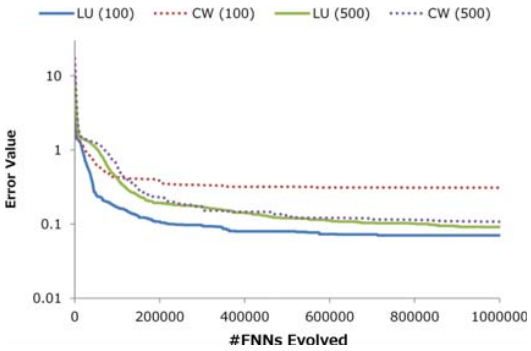

Fig. 15: Error value of the best FNN for each total number of FNNs evolved for modeling $F_3(x)$.

TABLE I: Number of repairs for genotype fuzzy numbers (averaged over 5 runs)

	$F_1(x)$	$F_2(x)$	$F_3(x)$
LU (100)	2.72E+6	2.13E+6	2.05E+6
LU (500)	2.73E+6	2.23E+6	2.40E+6
CW (100)	3.05E+6	2.56E+6	2.84E+6
CW (500)	2.77E+6	2.70E+6	2.61E+6

as genotype values.

The result that the LU model is better than the CW model is surprising because the search space is larger for the FES with the LU model than that for the FES with the CW model: $-10.0 \leq x^L_{j,i}, x^U_{j,i} \leq 10.0$ with the LU model, while $-10.0 \leq x^c_{j,i} \leq 10.0$ and $0.0 \leq x^w_{j,i} \leq 10.0$ with the CW model. To investigate the reason why the FES could evolve better FNNs with the LU model, the author counts the number of repairs for the invalid genotype values. As described in Section III.B, $x^U_{z,i}$ must not be smaller than $x^L_{z,i}$ in the LU model, while $x^w_{z,i}$ must not be negative in the CW model. In the reproduction process, if the new values of $x^L_{z,i}$, $x^U_{z,i}$, or $x^w_{z,i}$ violate the constraints, then the new values are repaired. Such repairs may interfere with the evolution of FNNs because the repairs restrict the modifications of weights and biases by the FES. Thus, fewer repairs will be better in the evolution of FNNs. Table I shows the number of repairs; the values in the table are the averaged values of five runs under each condition. For example, (10,100)-ES with the LU model required 2.72E+6 (2.72×10^6) repairs on average for $F_1(x)$, while (10,100)-ES with the CW model required 3.05E+6 (3.05×10^6) repairs on average for $F_1(x)$. Table I reveals that the LU model required fewer repairs than the CW model, which is the reason that the LU model could contribute for the FES to evolve better FNNs.

C. Comparison with Fuzzy Genetic Algorithm

Fig. 16 shows the error values of the best FNN for $F_1(x)$ in the same manner as Fig. 13, where "FES" shows the result obtained by using the FES proposed in this paper, while "FGA" shows the result obtained by using the FGA that the author previously proposed [6]. Figs. 17 and 18 show the error values for $F_2(x)$ and $F_3(x)$, respectively, in the same manner as Fig. 16.

These figures reveal that for all three target functions, the FES could evolve better FNNs than the FGA in the very early generations (i.e., before the total number of evolved FNNs reached approximately 100,000), but the FGA could evolve better after that. This result is attributed to the fact that the traditional reproduction operation of ES (which is employed in the proposed FES) contributes well to local exploitation but not to global exploration. The FES was likely to prematurely converge the population into a local minimum, while the FGA could explore the search space well by using the crossover and mutation operations. A promising idea for a more efficient search will be a combined application of the abovementioned approaches: the FES is applied in the former generations,

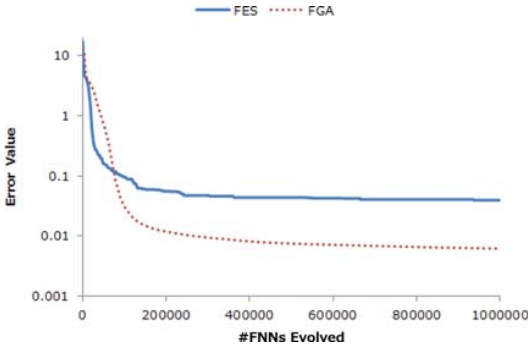

Fig. 16: Error value of the best FNN for each total number of FNNs evolved for modeling $F_1(x)$.

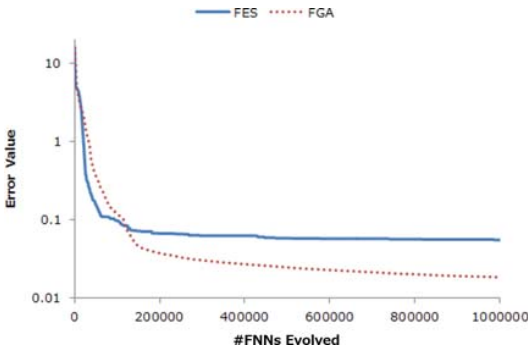

Fig. 17: Error value of the best FNN for each total number of FNNs evolved for modeling $F_2(x)$.

while the FGA is applied in the latter generations. Such a combination of ES and GA was found to be useful in literature [28].

V. Conclusion

In this paper, the author proposed a fuzzy-valued extension of ES and applied it to the neuroevolution of neural networks with fuzzy weights and biases. In the proposed FES, genotype values are not real numbers but fuzzy numbers. To handle the fuzzy genotype values, the FES extends its processes of initialization of population and reproduction.

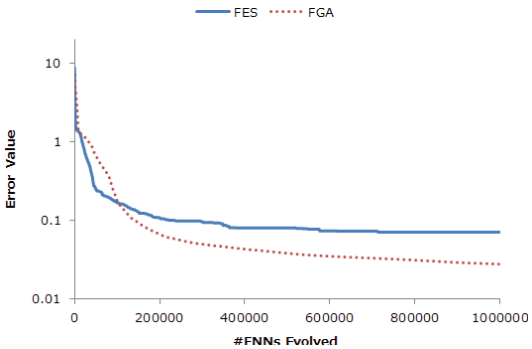

Fig. 18: Error value of the best FNN for each total number of FNNs evolved for modeling $F_3(x)$.

The FES was challenged to evolve FNNs that modeled each of the three hidden fuzzy functions. The experimental results showed that the best FNNs evolved by the FES approximated the target functions (particularly for larger membership scores) despite the fact that no training data were explicitly provided. In addition, the results revealed that the LU model contributed better to the FES than the CW model although the search space was larger for the FES with the LU model. This could be attributed to the fact that the FES with the LU model required fewer repairs for invalid genotype values than the FES with the CW model.

In the future, the author will further evaluate the ability of the FES by applying it to problems other than NE, e.g., evolving fuzzy if-then rules for fuzzy inference systems.

Acknowledgment

This research was supported by a Kyoto Sangyo University Research grant E1207.

References

[1] H. Ishibuchi, H. Tanaka, and H. Okada, Fuzzy neural networks with fuzzy weights and fuzzy biases, Proc. of IEEE International Conferences on Neural Networks, pp.1650-1655, 1993.
[2] D.B. Fogel, L.J. Fogel, and V.W. Porto, Evolving neural networks, Biological Cybernetics, vol.63, issue 6, pp.487-493, 1990.
[3] X. Yao, Evolving artificial neural networks, Proc. of the IEEE, vol.87, no.9, pp.1423-1447, 1999.
[4] K.O. Stanley and R. Miikkulainen, Evolving neural networks through augmenting topologies, Evolutionary Computation, vol.10, no.2, pp.99-127, 2002.
[5] D. Floreano, P. Durr, and C. Mattiussi, Neuroevolution: From architectures to learning, Evolutionary Intelligence, vol.1, no.1, pp.47-62, 2008.
[6] H. Okada, Genetic algorithm with fuzzy genotype values and its application to neuroevolution, International Journal of Computer, Information Science and Engineering, vol.8, no.1, pp.1-7, 2014.
[7] H-P. Schwefel, Evolution and Optimum Seeking, Wiley, 1995.
[8] H-G. Beyer and H-P. Schwefel, Evolution strategies - A comprehensive introduction, Natural Computing, vol.1, no.1, pp.3-52, 2002.
[9] N. Hansen and A. Ostermeier, Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation, Proc. of the 1996 IEEE International Conference on Evolutionary Computation, pp.312-317, 1996.
[10] M. Herdy, Evolution strategies with subjective selection, Proc. of the 4th International Conference on Parallel Problem Solving from Nature, pp.22-31, 1996.
[11] J. Knowles and D. Corne, The pareto archived evolution strategy: A new baseline algorithm for pareto multiobjective optimisation, Proc. of the 1999 Congress on Evolutionary Computation, pp.98-105, 1999.
[12] N. Hansen and A. Ostermeier, Completely derandomized self-adaptation in evolution strategies, Evolutionary Computation, vol.9, no.2, pp.159-195, 2001.
[13] H.G. Beyer, The Theory of Evolution Strategies, Springer, 2001.
[14] A. Auger and N. Hansen, A restart CMA evolution strategy with increasing population size, Proc. of the 2005 IEEE Congress on Evolutionary Computation, pp.1769-1776, 2005.
[15] O.M. Shir and T. Back, Niching in evolution strategies, Proc. of the 2005 Conference on Genetic and Evolutionary Computation, pp.915-916, 2005.
[16] N. Hansen, The CMA evolution strategy: A comparing review, Towards a New Evolutionary Computation, Springer, pp.1769-1776, 2006.
[17] D.V. Arnold, Weighted multirecombination evolution strategies, Theoretical Computer Science - Foundations of Genetic Algorithms, vol.361, no. 1, pp.18-37, 2006.

[18] X. Chen, X. Liu, and Y. Jia, Combining evolution strategy and gradient descent method for discriminative learning of bayesian classifiers, Proc. of the 11th Annual Conference on Genetic and Evolutionary Computation, pp.507-514, 2009.
[19] D.V. Arnold and A.S. Castellarin, A novel approach to adaptive isolation in evolution strategies, Proc. of the 11th Annual Conference on Genetic and Evolutionary Computation, pp.491-498, 2009.
[20] L. Graening, N. Aulig, and M. Olhofer, Towards directed open-ended search by a novelty guided evolution strategy, Proc. of the 11th International Conference on Parallel Problem Solving from Nature: Part II, pp.71-80, 2010.
[21] A. Auger, D. Brockhoff, and N. Hansen, Analyzing the impact of mirrored sampling and sequential selection in elitist evolution strategies, Proc. of the 11th Workshop on Foundations of Genetic Algorithms, pp.127-138, 2011.
[22] A. Auger, D. Brockhoff, and N. Hansen, Mirrored sampling in evolution strategies with weighted recombination, Proc. of the 13th Annual Conference on Genetic and Evolutionary Computation, pp.861-868, 2011.
[23] I. Loshchilov, M. Schoenauer, and M. Sebag, Self-adaptive surrogate-assisted covariance matrix adaptation evolution strategy, Proc. of the Fourteenth International Conference on Genetic and Evolutionary Computation Conference, pp.321-328, 2012.
[24] J. Lu, B. Li, and Y. Jin, An evolution strategy assisted by an ensemble of local gaussian process models. Proc. of the Fifteenth Annual Conference on Genetic and Evolutionary Computation Conference, pp.447-454, 2013.
[25] R. Li, M.T.M. Emmerich, J. Eggermont, T. Back, M. Schutz, J. Dijkstra, and J.H.C. Reiber, Mixed integer evolution strategies for parameter optimization, Evolutionary Computation, vol.21, no.1 pp.29-64, 2013.
[26] L.A. Zadeh, The concept of a linguistic variable and its application to approximate reasoning - I, II, and III, Information Sciences, vol.8, pp.199-249, pp.301-357, and vol.9, pp.43-80, 1975.
[27] G. Alefeld and J. Herzberger, Introduction to Interval Computation, Academic Press, 1983.
[28] H. Okada, J. Tokida, and Y. Fujii, Comparison of evolution strategy, genetic algorithm and their hybrids on evolving autonomous game controller agents, International Journal of Science and Engineering Investigations, 10612-02, vol.1, no. 6, pp.11-16, 2012.

Hidehiko Okada is currently Professor with the Department of Computer Science and Engineering, Kyoto Sangyo University, Kyoto, Japan. He received his B.S. in Industrial Engineering and Ph.D. in Engineering from Osaka Prefecture University in 1992 and 2003, respectively. He was a researcher with NEC Corporation from 1992 to 2003, and since 2004, he has been with Kyoto Sangyo University. His current research interests include computational intelligence and human-computer interaction. He is a member of Information Processing Society of Japan, Institute of Electronics, Information and Communication Engineers, Society of Instrument and Control Engineers, Japanese Society for Artificial Intelligence, Japan Society for Fuzzy Theory and Intelligent Informatics, and Human Interface Society.

He received the annual Best Paper award by Journal of the Institute of Industrial Applications Engineers in 2013.