Modeling Nonlinear Systems by a Fuzzy Logic Neural Network Using Genetic Algorithms

Abdel-Fattah Attia, P. Horáček

The main aim of this work is to optimize the parameters of the constrained membership function of the Fuzzy Logic Neural Network (FLNN). The constraints may be an indirect definition of the search ranges for every membership shape forming parameter based on 2^{nd} order fuzzy set specifications. A particular method widely applicable in solving global optimization problems is introduced. This approach uses a Linear Adapted Genetic Algorithm (LAGA) to optimize the FLNN parameters. In this paper the derivation of a 2^{nd} order fuzzy set is performed for a membership function of Gaussian shape, which is assumed for the neuro-fuzzy approach. The explanation of the optimization method is presented in detail on the basis of two examples.

Keywords: genetic algorithms, fuzzy logic neural network, 2^{nd} order fuzzy sets.

1 Introduction

A fuzzy logic neural network (FLNN) [10] is a general nonlinear interpolator used for modeling static systems. The structure and rough setting of FLNN parameters is usually done manually by an expert. Fine-tuning is done by numerical optimization techniques using reference input-output data. The use of a random search technique is an option for solving the problem. Genetic algorithm (GA) is an evolutionary method that simulates the process of natural selection and survival of the fittest. GAs randomly generate a set of potential problem solutions, and manipulate them using genetic operators. Each solution is assigned a scalar fitness value, which is a numerical assessment of how well it solves the problem. Through crossover and mutation operations new feasible solutions are hopefully generated. The process continues until the termination condition is met. Further discussion on GAs can be found in [6], [17], and [5]. When GA is implemented as a learning procedure, the FLNN parameters are coded to form a string referred to as a chromosome. Under instances in the population of chromosomes, the genetic operations are performed. The fitness is inversely proportional to the whole system error, which represents the difference between the required and actual network response.

The remainder of this paper is organized as follows: Section 2 illustrates the structure of a Fuzzy Logic Neural Network model. In section 3, the derivation of the membership function (*MF*) constraints is performed for *MFs* of Gaussian shape. Section 4 shows the LAGA approach. The proposed genetic algorithm with constrained search space is explained in detail in section 5. The explanation of the optimization method is presented in detail on the basis of two application examples, and the conclusion is presented in Section 6.

2 Proposed fuzzy logic neural network (FLNN)

The FLNN model as a general nonlinear interpolator is built using the multilayer fuzzy logic neural network shown in Fig. 1, proposed by Lin [10], with some modification by [9]. This is a particular implementation of a fuzzy system equipped with fuzzification and defuzzification interfaces. This network represents a linguistic fuzzy system with a general rule-based structure. The following example demonstrates this structure:

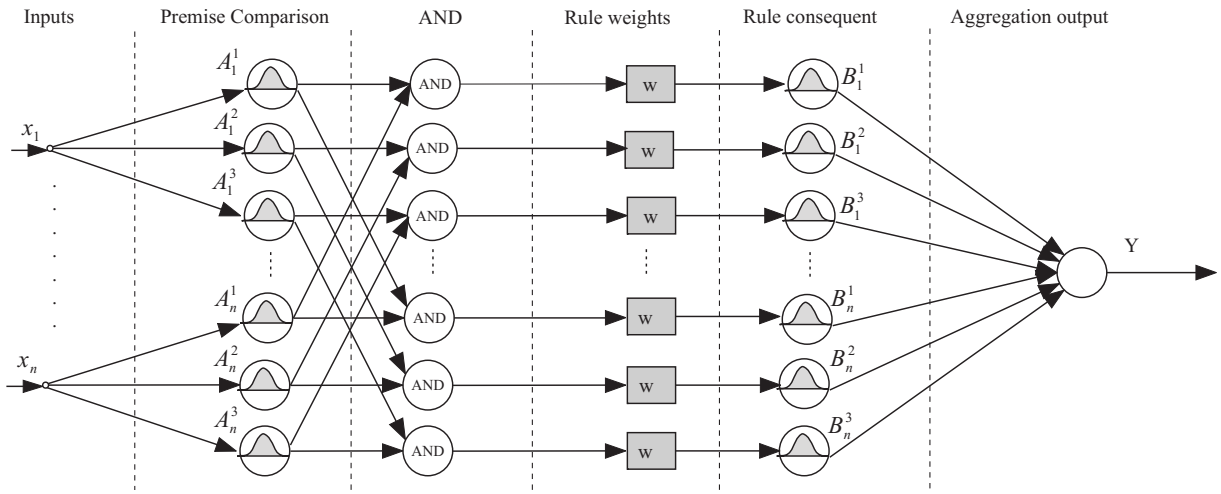


Fig. 1: Fuzzy logic neural network topology

An FLNN consists of several layers [7], [9]:

Layer 1: Actual values of the input variables are stored in this layer. Generally, fuzzy sets are considered as the input values (crisp numbers are special cases of fuzzy sets). The fuzzy sets are in parametric form or in look-up table form.

Layer 2: Rule premises (input reference fuzzy sets) are stored here. The actual input value is compared with the rule premise using degree of overlapping:

$$D_T(X, A) = \sup_x T\big(X(x), A(x)\big) = \mathrm{hgt}(X \cap_T A) \tag{1}$$

where T is the selected t-norm and *hgt* is the height of intersection of X and A with respect to t-norm T. In the special case of crisp input $X = x^*$ the $D_T(X, A)$ is simply

$$D_T\big(X, A\big) = A(x^*) \tag{2}$$

For some parametric fuzzy sets and some t-norms for $D_T(X, A)$ an analytical expression can be derived. For the other cases it must be computed numerically.

Layer 3: Every neuron in this layer performs a fuzzy conjunction using the selected t-norm.

$$y = T\big(u_1, u_2, \ldots, u_n\big) \tag{3}$$

The common parameter of the layer is the type of t-norm and its parameters.

Layer 4: Every neuron represents a rule weight w. The output of the neuron is the overall degree of rule activation *act,* and is computed as follows:

$$y = act = w \cdot u \tag{4}$$

where parameter w has to lie in the interval [0,1].

Layer 5: Only rule consequents (output reference fuzzy sets) are stored in this layer. The fuzzy sets are usually in the parametric form. The input of the neuron is the overall degree of rule activation *act*. This value is attached to the reference fuzzy set, and together they are fed to the next aggregation layer.

Layer 6: The output of the network is computed here, using the selected aggregation (inference) algorithm. There is a corresponding fuzzy set B_i with its activation degree act_i in the i-th input of each neuron. When we use the Mamdani inference algorithm, the output fuzzy set is computed as follows:

$$Y(y) = \max_{i=1}^{n} T\big(act_i, B_i(y)\big) \tag{5}$$

where n is the number of inputs to the neuron.

When we use a fuzzy arithmetic based inference algorithm, the output fuzzy set is computed as follows:

$$Y_i = \sum_{i=1}^{n} act_i \cdot B_i \Big/ \sum_{i=1}^{n} act_i \tag{6}$$

Usually only crisp output value y is needed. Then we use a defuzzification method to get the crisp value. The most widely used method is centroid average defuzzification:

$$Y = \sum_{i=1}^{n} act_i \cdot y_i \Big/ \sum_{i=1}^{n} act_i \tag{6}$$

where y_i is a centroid of fuzzy set B_i.

FLNN works in the following manner [7], [9]. In the forward run the input values (crisp values, fuzzy sets) are first compared with all premises of the rules (input reference fuzzy sets). The outputs of the AND-neuron are then combined with rule-weight (preference between rules) to obtain the degree of rule activation. In the last layer these degrees are aggregated with the corresponding consequents of the rules (output reference fuzzy sets) according to the inference algorithm. The output of the FLNN can be a fuzzy set or a crisp value (after defuzzification).

3 Determining constraints of MF parameters

In the case of FLNN the membership functions $MF_j(x_i)$ of input x_i and output y are frequently approximated by Gaussians. A Gaussian shape is formed by two parameters: mathematical expectation c and standard deviation σ as in formula (8):

$$MF_j(x_i) = G_j\big(x_i, c_j, \sigma_j\big) = e^{\left(-\frac{(x_i - c_j)^2}{2\sigma_j^2}\right)} \tag{8}$$

The idea of 2nd order fuzzy set was introduced to get the boundary of Gaussian shape of membership function by [11] . The 2nd order fuzzy set of a given MF (x) is the area between d^+ and d^-, where d^+, and d^- are the upper and the lower crisp boundaries of the 2nd order fuzzy sets, respectively, as shown in Figure 2 [2]. The expressions to determine its crisp boundaries are (9), and (10):

$$d_j^+(x_i) = \min\big(1, MF_j(x_i) + \delta\big) \tag{9}$$

$$d_j^-(x_i) = \max\big(0, MF_j(x_i) - \delta\big) \tag{10}$$

Formula (9) and formula (10) are based on the assumptions that the height of the slice of the 2nd order fuzzy region, bounded by d^+ and d^- , at point x is equal to 2δ where $\delta \in [0, 0.3679]$ and these boundaries are equidistant from MF (x). To obtain the ranges for the shape forming parameters of the MFs, it should be assumed that these 2nd order fuzzy sets are MF search spaces. Therefore, all MFs with acceptable parameters should be inside the area. In the general case the intervals of acceptable values for every MF shape forming parameter (e.g., $\Delta c = [c_{11}, c_{22}]$, and $\Delta\sigma = [\sigma_{11}, \sigma_{22}]$ for Gaussians) may be determined by solving formulas (8), (9), and (10). In practice, this may be done approximately considering d^+ and d^- as soft constraints. For instance, c_{11} and c_{22} for Gaussians may be found as the maximum root and the minimum root of the equation $d^+ = 1$, which can easily is to be calculated. This equation is based on the assumption that a fuzzy set represented by the Gaussian must have a point where it is absolutely true. σ_{11}, and σ_{22} can easily be found from the following four equations:

$$\begin{aligned} &G\big(c + \sigma, c, \sigma\big) + \delta = G\big(c + \sigma, c, \sigma_{22}\big); \text{ and} \\ &G\big(c + \sigma, c, \sigma\big) - \delta = G\big(c + \sigma, c, \sigma_{11}\big) \end{aligned} \tag{11}$$

$$\begin{aligned} &G\big(c - \sigma, c, \sigma\big) + \delta = G\big(c - \sigma, c, \sigma_{22}\big); \text{ and} \\ &G\big(c - \sigma, c, \sigma\big) - \delta = G\big(c - \sigma, c, \sigma_{11}\big) \end{aligned} \tag{12}$$

where we choose σ_{11} as minimum and σ_{22} as maximum from the roots. These equations are based on the assumption that the acceptable Gaussians with $[\sigma_{11}, \sigma_{22}]$ should cross the 2nd order fuzzy region slices at points $x = c \pm \sigma$. There are two options finding the constraints of Gaussian parameters.

First, considering constraints as a hard constraint and it follows: the lower and upper bounds of the center of the Gaussian membership function will be chosen as c_{min}, and c_{max} should be less than the values of c_{11} and c_{22} to satisfy the search space constraint conditions of 2nd order fuzzy sets as shown in Fig. 3. The lower, and upper bounds for spread of Gaussian membership function σ_{min}, and σ_{max} will be equal to σ_{11} and σ_{22}, respectively to satisfy search space constraint conditions of 2nd order fuzzy sets, as shown in Fig. 2. A second option is to consider these constraints as a soft constraint, i.e., [c_{min}, c_{max}] equal [c_{11}, c_{22}], and [σ_{min}, σ_{max}] equal [σ_{11}, σ_{22}].

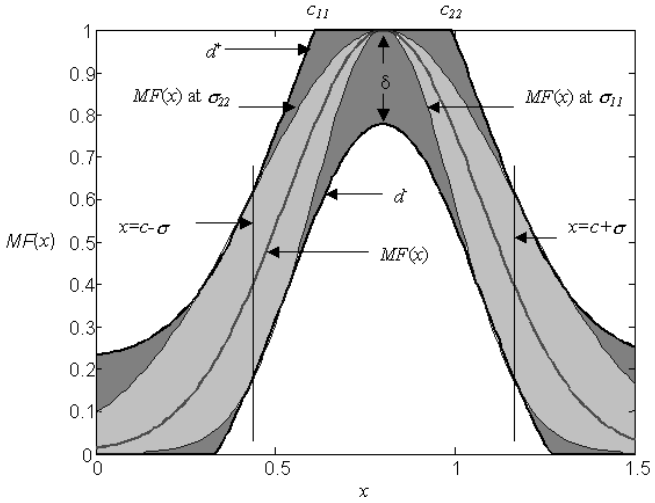


Fig. 2: Upper and lower boundaries of spread, σ using a 2nd order fuzzy set

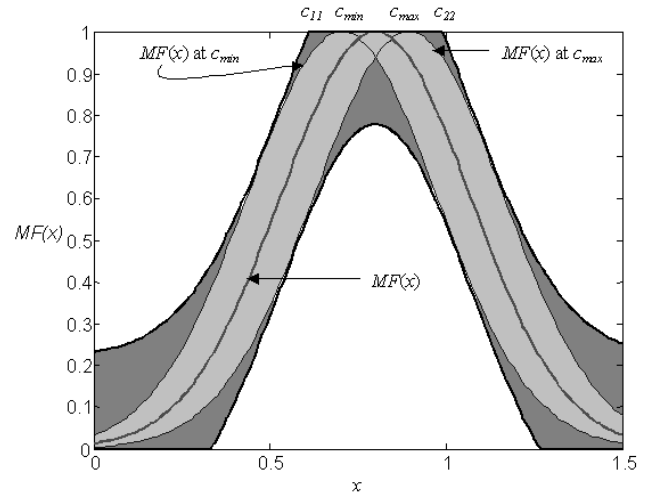


Fig. 3: Upper and lower boundaries of center, c using a 2nd order fuzzy set

4 Linear Adapted Genetic Algorithm (LAGA)

For tuning MF parameters a particular evolutionary algorithm a genetic algorithm is chosen. Varying the crossover probability rate P_c and mutation probability rate P_m the GA's control parameters provide faster convergence than constant probability rates. P_c is set high at the beginning of the generation and decreases linearly with generations as in (13) [1]. As is known from the Standard Genetic Algorithm (SGA), at the beginning of generation the randomized initial GA population is diverse. This means that promising solutions are scattered through the search space. So, P_c is high in the initial generation, but over the generations these solutions will generate even better solutions. This means that the population converges to a smaller subset of the search space, and the P_c value will decrease according to formula (13), where there are large values for $P_c(0.5-1.0)$, and small values of $P_m(0.0-0.005)$ [6], [13], and [3].

$$P_c(x) = \frac{-0.5x}{M-1}(x-1)+1, \quad x \in \{1, M\}, \qquad (13)$$

where x is the number of generations, and M is the maximum of generations allowed. As is known, mutation is not needed at the beginning of generation, where the members of the population are very distinct. The value of P_m increases linearly as a function of the number of generations to exploit the improved solution in the established region of the current best solution. This is clear from equation (14).

$$P_m(x) = \frac{0.005}{M-1}(x-1), \quad x \in \{1, M\}. \qquad (14)$$

5 Proposed genetic algorithm with constrained search space

The main aspects of the proposed LAGA for optimizing FLNN are discussed below and the block diagram for the LAGA optimization process is shown in Fig. 5.

5.1 Fuzzy model representation

This section discusses how the proposed FLNN is formulated using the LAGA approach, where all the parameters of the FLNN are represented in a chromosome. The chromosome representation determines the GA structure. With a population size (*popsize*), we encode the parameters of each fuzzy model in a chromosome, as a sequence of elements describing the input fuzzy sets in the rule antecedents followed by the parameters of weights and the rule consequents. Where the intervals of acceptable values for every *MF* shape forming parameter ($\Delta c = [c_{min}, c_{max}]$, and $\Delta\sigma = [\sigma_{min}, \sigma_{max}]$ for Gaussians) are determined based on 2nd order fuzzy sets for all membership functions, as explained in section 4. The acceptable constraints for rule weights are between [0,1], and for centroids they are the minimum and maximum values of the output.

5.2 Coding of FLNN parameters

Fig. 1 shows n inputs (x_1, x_2, …, x_n) and one output y. Each of the input fuzzy variables is classified into m reference fuzzy sets. Every reference fuzzy set is described by a Gaussian membership function specified by two parameters: center c and spread σ, resulting in $(2 \times m \times n)$ parameters at the corresponding layer. Using the Wang technique for generating rules from given data [16], the fuzzy model has k rules from $(m)^n$ rules theoretically possible. This means have k rule weights; w and k centroids represented by singletons b. Thus a total of $2(m \times n+k)$ parameters $(2 \times m_{membership\_functions} \times n_{variables} + k_{weights} + k_{centroids})$ need to be optimized using LAGA. The coded parameters of

Table 1: Coded parameters of FLNN

Chromosome	Sub-chromosome of inputs	Sub-chromosome of rule weights	Sub-chromosome of rule consequents
	x_1,……, x_n	w_1, ……, w_k	b_1,……, b_k
Parameters $2(m \times n + k)$	c_1, σ_1, ……, c_n, σ_n $(2m \times n)$	w_1, ……, w_k k	b_1, ……, b_k k
Gene	1000000000…0110100011	0100111111…	0111001010 …

FLNN are arranged as shown in Table 1 to form the chromosome of the population.

5.3 Selection function

The selection strategy decides how to select individuals to be parents for new '*Childs*'. Usually the selection applies some selection pressure by favoring individuals with better fitness. After procreation, the suitable population consists for example of L chromosomes, which are all initially randomized. Each chromosome has been evaluated and associated with fitness, the current population undergoes the reproduction process to create the next population, and the "roulette wheel" selection scheme is used to determine the member of the new population. The chance on the roulette-wheel is adaptive and is given as $P_l / \sum P_l$, where

$$P_l = \left(\frac{1}{J_l} \right), \quad l \in \{1, \ldots, L\}$$

and J_l is the performance of the model encoded in the chromosome measured in terms of the normalized *Root Mean Square Error* (*RMSE*):

$$J(\theta) = \frac{\sum_{i=1}^{N} \left\| y\left(x^i, \theta\right) - \hat{y}^i \right\|^2}{\sum_{i=1}^{N} \left\| \hat{y}^i - \bar{y} \right\|^2} \qquad \hat{y}$$

where N is the number of point samples, θ is the required parameters to be optimized, y is the true output, $\bar{y} = \frac{1}{N} \sum_{i=1}^{N} y$,

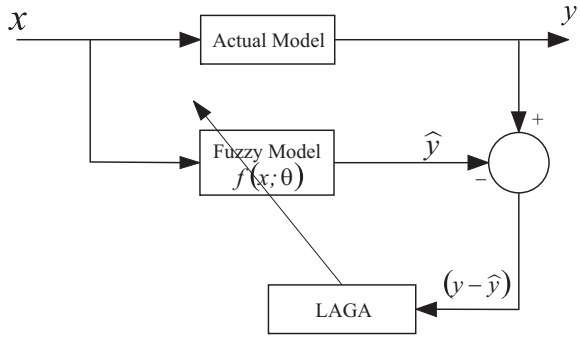


Fig. 4: Block diagram of parameters identification

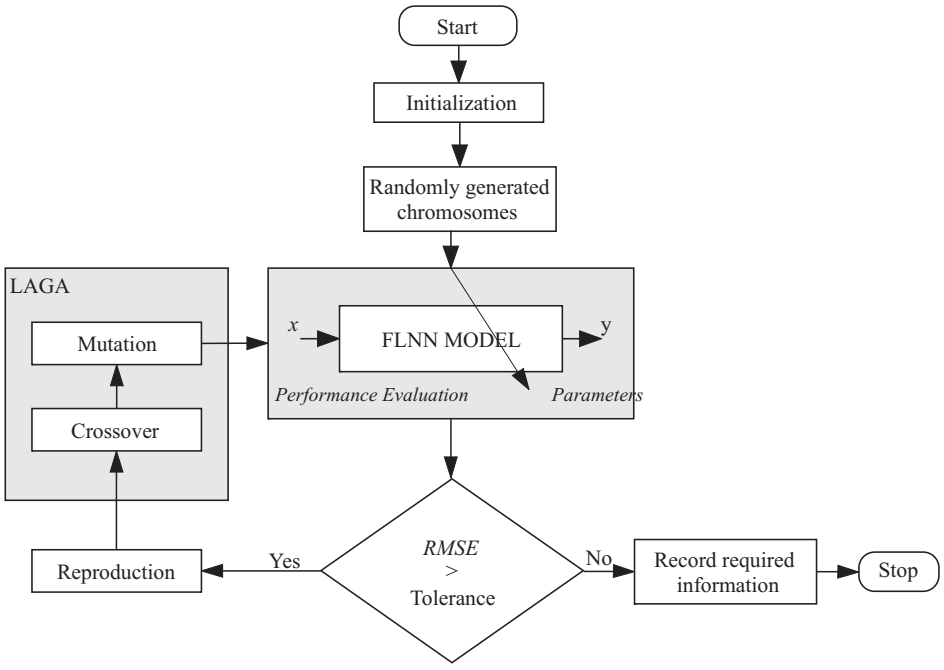


Fig. 5: Block diagram for the LAGA optimization process

and $\hat{y}$ is the model output, as shown in Fig. 4. The inverse of the selection function is used to select chromosomes for deletion.

5.4 Crossover and mutation operators

The mating pool is formed, and crossover is applied and followed by a mutation operation following the LAGA approach. Finally, after these three operations, the overall fitness of the population is improved. The procedure is repeated until the termination condition is reached. The termination condition is the maximum allowable number of generations, or a certain value of (*RMSE*) required to be reached.

6 Applications

6.1 Modeling the Mackey-Glass process

The process used as an object of modeling is defined by the chaotic Mackey-Glass differential delay equation [8]:

$$x(t) = \frac{0.2\, x(t-\tau)}{1 + x^{10}(t-\tau)} - 0.1\, x(t) \qquad (15)$$

The prediction of future values of this time series is a benchmark problem, which has been considered by a number of connectionist researchers. A time window of the process behavior is shown in Fig. 6. The sampling period used in the numerical study is set to 0.1, initial condition $x(0) = 1.2$ and time delay $\tau = 17$. In according with [8], we use the samples of $x(t-18)$, $x(t-12)$, $x(t-6)$ and $x(t)$ to predict $x(t+6)$.

For the chaotic system, the model has *four* input variables $x(t-18)$, $x(t-12)$, $x(t-6)$ and $x(t)$, and a single output $x(t+6)$. The values of every input variable are classified into *three* reference fuzzy sets. Every reference fuzzy set is described by a Gaussian membership function specified by two parameters: center c and spread σ, resulting in 24 parameters for inputs. Using the Wang technique for generating rules from the given data [16], we have 25 rules from 81 rules theoretically possible. This means we have 25 rule weights w and 25 centroids represented by singletons b. Thus a total of 74 parameters ($2 \times 3_{\text{membership\_function}} \times 4_{\text{variables}} + 25_{\text{weights}} + +25_{\text{centroids}}$) need to be optimized using LAGA. The coded parameters of FLNN are arranged as shown in Table 1 to form the chromosome of the population. To describe the LAGA optimization process, consider the block diagram shown in Fig. 5. At the beginning of the process, the initial population comprises a set of chromosomes. Every chromosome has 74 genes, and every gene has 10 bits, so the chromosome length is 740 bits.

Simulation Results

From the Mackey-Glass time series $x(t)$ (15), we extracted 3000 input-output pairs. The first 1000 data samples were used to build the fuzzy model, while the remaining 2000 data samples were used for model testing. Fig. 7 depicts the corresponding membership functions before and after training using LAGA. There were 420 generations, and 60 minutes of computation time using MATLAB and PC 400 MHz with 64 MB of RAM. Fig. 8 shows that the FLNN model follows the actual process. The MSE is 0.0009 and the RMSE is 0.14 as shown in Fig. 9.

Figure 10 shows the shape of constraints of membership functions based on second order fuzzy sets, as explained in section 3 with δ=0.3.

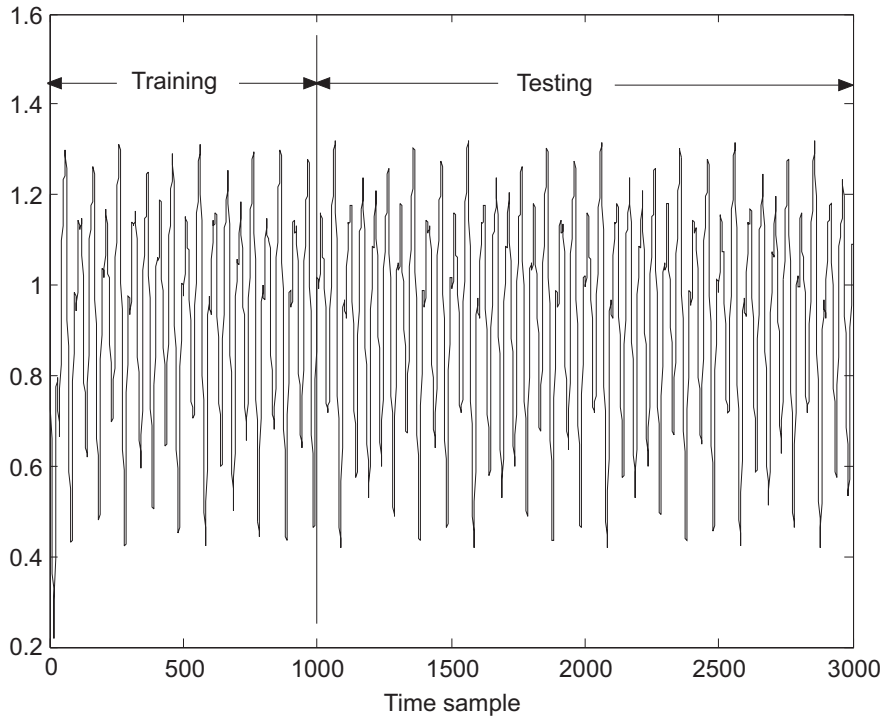


Fig. 6: Mackey-Glass time series

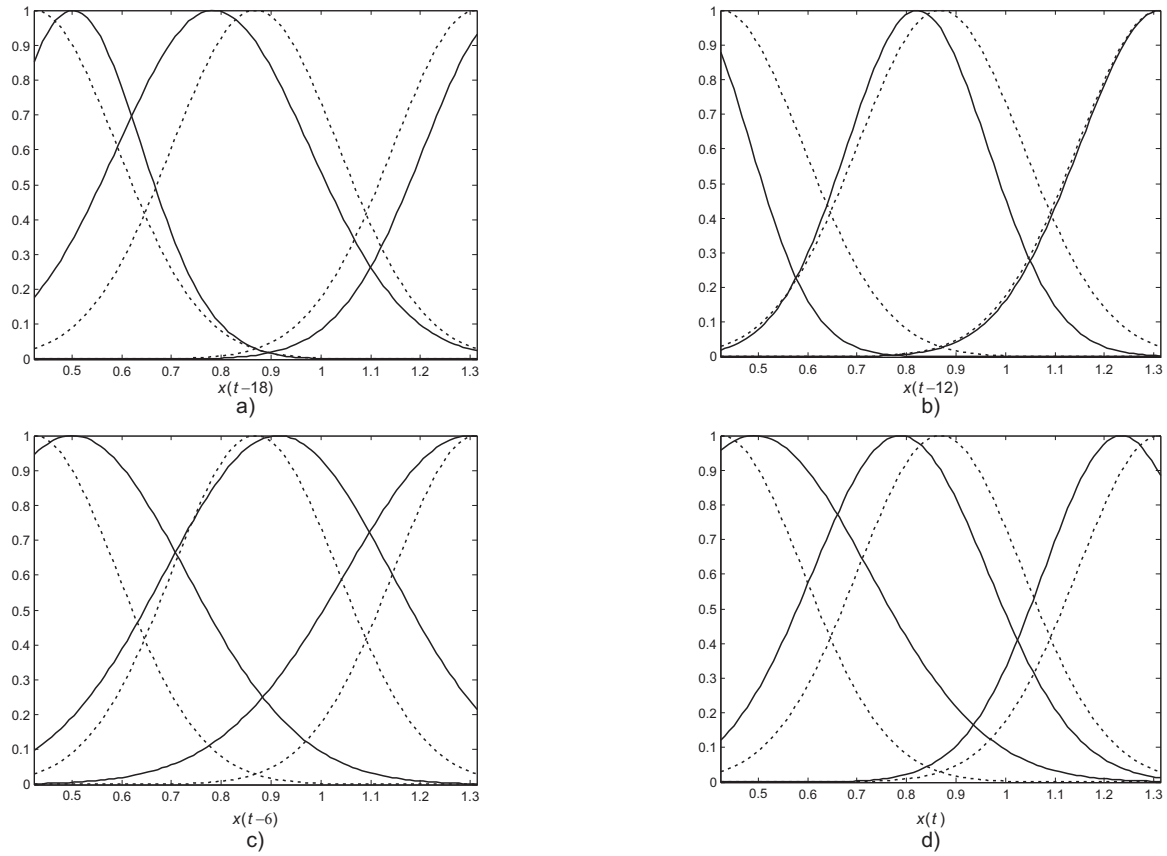


Fig. 7: Membership functions of reference fuzzy sets for inputs a) $x(t-18)$, b) $x(t-12)$, c) $x(t-6)$, and d) $x(t)$ (Dashed line: Normalized MFs before learning. Solid line: Optimized MFs after using LAGA)

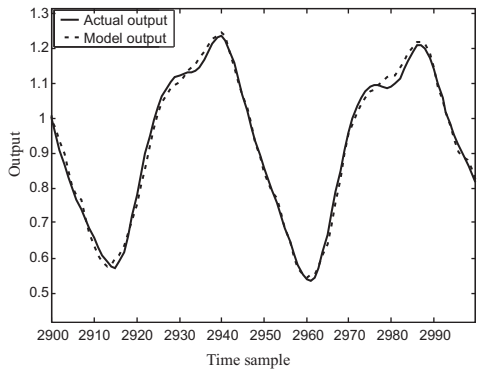


Fig. 8: Comparing the actual process with fuzzy model after training

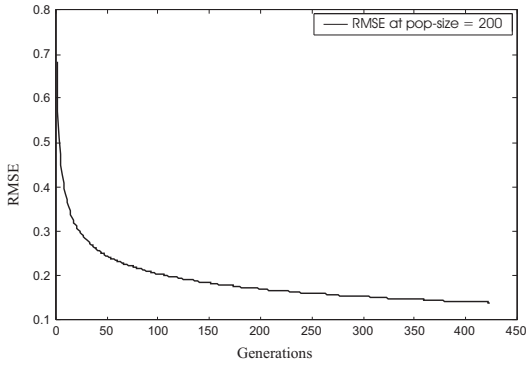


Fig. 9: LAGA convergence rate at population size 200

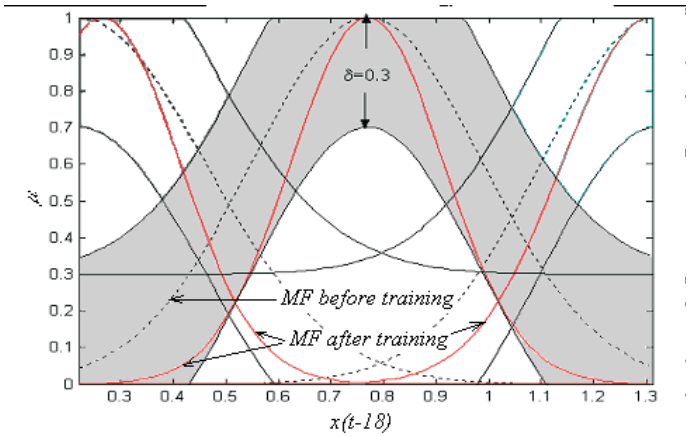


Fig. 10: Membership functions of reference fuzzy sets for inputs $x(t-18)$. (Dashed line: Normalized MFs before learning. Solid line: Optimized MFs after using LAGA)

6.2 Nonlinear discrete time process modeling and identification

This example is taken from [12], [15], in which the plant to be identified is governed by the differential equation (19):

$$y(k+1) = 0.3(k) + 0.6(k-1) + g[u(k)] \qquad (19)$$

where the unknown function has the form

$g(u) = 0.6\sin(\pi u) + 0.3\sin(3\pi u) + 0.1\sin(5\pi u)$.

The plant is modeled using FLNN, as described in section 2. The model has three input variables $u(k)$, $y(k)$, and $y(k-1)$ and a single output $y(k+1)$, treated as linguistic variables. The universe of discourse of every variable is partitioned into five fuzzy sets with symmetrical Gaussian membership functions. There are 30 parameters at the input of the FLNN model. Using the Wang technique for generating rules from the given data, we have 20 rules. This means we have 20 weights, and 20 centroids represented by singletons. Thus a total of 70 parameters ($2 \times 5_{\text{membership\_functions}} \times 3_{\text{variables}} + 20_{\text{weights}} + 20_{\text{centroids}}$) need to be optimized using LAGA. The learning procedure of LAGA is applied as in the first numerical example. Fig. 5 shows the block diagram for the LAGA optimization process for optimizing the FLNN model parameters of the second numerical example. The process starts with zero initial conditions. The first 250 data points are used to build the fuzzy model at , while the remaining 450 data points are used to identify an FLNN model. As explained in the first application we determined the constraints for this application based on second order fuzzy sets with δ=0.28. Fig. 11 shows that the FLNN model has a good match with the actual model, with an MSE of 0.0473, and an RMSE of 0.0607, as shown in Fig. 12. Fig. 13 depicts the membership functions for each input variable before and after training using LAGA. Fig. 14 shows the output model and the plant for the input:

$u(k) = 0.5\sin(2\pi k/250)$ for $1 \le k \le 250$ and $501 \le k \le 700$,

and

$u(k) = 0.5\sin(2\pi k/250) + 0.5\sin(2\pi k/25)$ for $251 \le k \le 500$.

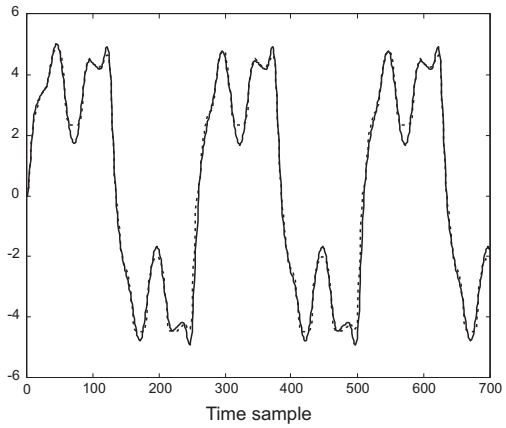


Fig. 11: Outputs of the plant (solid line), and the FLNN model (dashed line) for $u(k) = \sin(2\pi k/250)$

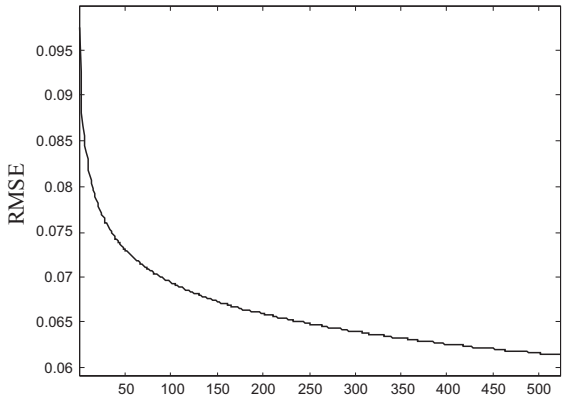
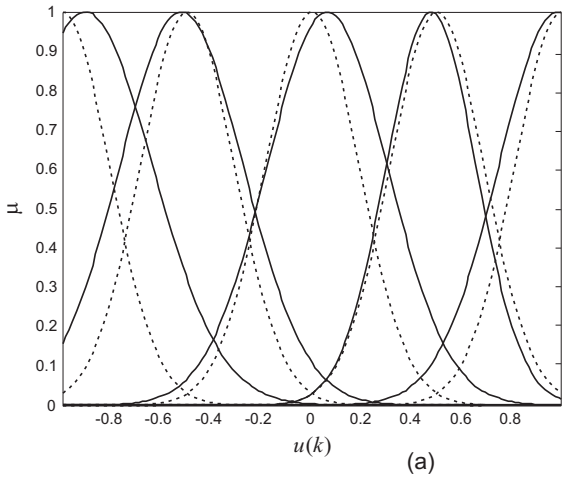
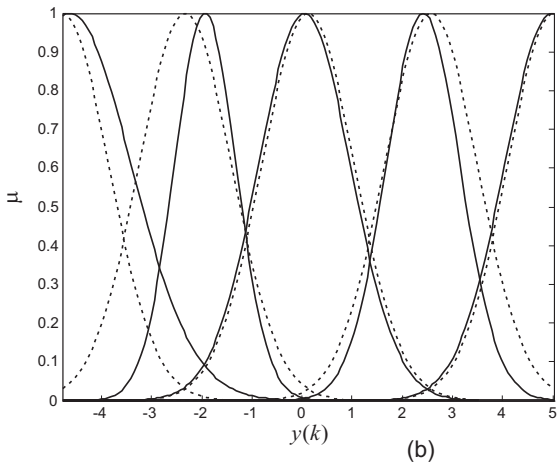


Fig. 12: LAGA convergence rate at population size 200





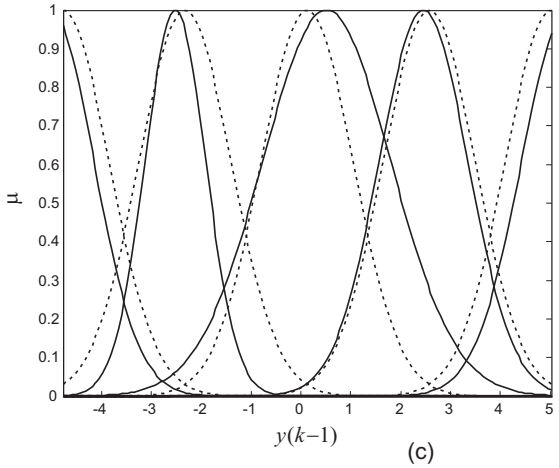


Fig. 13: (a, b, and c) are input MFs:
Dashed line: Normalized MFs before learning
Solid line: Optimized MFs after using LAGA

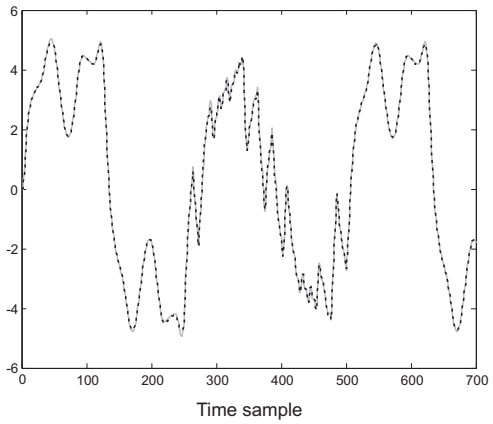


Fig. 14: Outputs of the plant (solid gray line), and FLNN model (dashed line) for the input $u(k)=0.5\sin(2\pi k/250)$ for $1 \le k \le 250$ and $501 \le k \le 700$, and $u(k)=0.5\sin(2\pi k/250)+0.5\sin(2\pi k/25)$ for $251 \le k \le 500$

7 Conclusion

The paper deals with modeling of nonlinear systems and processes using fuzzy logic neural networks. Reference data driven identification of parameters of fuzzy logic neural networks utilizing genetic algorithms has been proposed and tested. Specification of parameter constraints related to input reference fuzzy sets are based on 2nd order fuzzy sets. The problem of constrained nonlinear optimization is solved based on a genetic algorithm with variable crossover and mutation probabilities rates, LAGA [Attia, 2001]. The paper reports the results in dynamic process identification, prediction of time series in particular. The performance of the nonlinear models for time series prediction is examined. The simulation results of the application examples indicate the effectiveness of the proposed LAGA approach as a promising learning algorithm.

Acknowledgement

This work received support from the Ministry of Education of the Czech Republic under Project LN00B096.

References

[1] Attia, A.: *Global Optimization Method for Neuro-Fuzzy Modeling and Identification.* Research Report 335/01/203, CTU, Faculty of Electrical Engineering, Department of Control Engineering, Prague, 2001, p. 41

[2] Attia, A., Horáček, P.: *An Optimal Design Of a Fuzzy Logic Neural Network Using a Linear Adapted Genetic Algorithm*. In: 7th International Mendel Conference On Soft Computing, Brno, 2001, pp. 42–49

[3] Attia, A., Horáček, P.: *Adaptation Of Genetic Algorithms For Optimization Problem Solving*. In: 7th International Mendel Conference On Soft Computing, Brno, 2001, pp. 36–41

[4] Farag, W., Victor, H.: *A Genetic-Based Neuro-Fuzzy Approach for modeling and Control of Dynamical Systems.* IEEE Trans. On Neural Networks. Vol. 9, No. 5, September 1998

[5] Gen, M., Cheng, R.: *Genetic Algorithms and Engineering Optimizations*. John Wiley & Sons, 2000

[6] Goldberg, D.: *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison –Wesley, 1989

[7] Horáček, P.: *Fuzzy Modeling and Control.* In: H. Adelsberger, J. Lažanaský, V. Ma ík (eds.): *Information Management in computer Integrated Manufacturing*. Lecture Notes in Computer Science No. 973, Springer-Verlag Berlin, 1995, pp. 257–288

[8] Jang, S. R., Sun, C. T., Mizutani, E.: *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*. Prentice Hall, Inc., 1997

[9] Kolínský, J.: *Identifikace parametr fuzzy−logických neuronových sítí.* Diplomová práce, Katedra řídicí techniky, Fakulta elektrotechnická, ČVUT, 2000

[10] Lin, C. T., Lee, C.S.G.: *Neural–Network-Based Fuzzy Logic Control and Design Decision System*. IEEE Trans. on Computers, Vol. 40, No. 12/1991, pp. 1320–1336

[11] Melikhov, A., Miagkikh, V., Topchy, P.: In: *Optimization of Fuzzy and Neuro-Fuzzy Systems by means of Adaptive Genetic Search.* Proc. of GA+SE'96 IC, Gursuf, Ukraine, 1996

[12] Narendra, K. S., Parthasarathy, K.: *Identification and control of dynamical systems using neural networks*. IEEE Trans. Neural Networks, Vol. 1, 1990

[13] Srinivas, M., Patnaik, L. M.: In: *Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms.* IEEE Trans. System. Man. and Cybernetics, Vol. 24, No. 4/1994, pp. 656–667

[14] Srinivas, M., Patnaik, L. M.: In: *Genetic Search: Analysis Using Fitness Moments*. IEEE Trans. on Knowledge and Data Engineering. Vol. 8, No. 1/1996

[15] Wang Li-Xin: In: *Adaptive Fuzzy Systems and Control, Design and Stability Analysis*. by PTR Prentice Hall, 1994

[16] Wang Li-Xin, Mendel, J.: In: *Generating Fuzzy Rules by Learning from Examples.* IEEE Trans. Systems, Man, and Cybernetics, Vol. 22, No. 6/1992, pp. 1414–1427

[17] Winter, G., Périanx, J., Mgalán, Cuesta, P.: *Genetic Algorithms in Engineering and Computer Science*. ISBN 04-71-95859-X, John Wiley & Sons, 1996

Ing. Abdel Fattah Attia, M.Sc.
phone: +420 2 2435 7612
fax: +420 2 2435 7298
e-mail: attiaa1@control.felk.cvut.cz

Department of Control Engineering

Doc. Ing.,Petr Horáček, CSc.
phone: +420 2 2492 2241
e-mail: horacek@control.felk.cvut.cz

Center for Applied Cybernetics

Czech Technical University in Prague
Faculty of Electrical Engineering
Technická 2,
166 27 Praha 6, Czech Republic

