Inference method and tuning of a neuro-fuzzy system with fuzzy inputs on a graphic processor

Vasiliy G. Sinuk [1], Maxim V. Panchenko [1]

[1] *Belgorod State Technological University named after V.G. Shouhov, Kostyukov str. 46, Belgorod, 308012, Russia*

Abstract

The paper deals with the learning process of Mamdani-type fuzzy systems with fuzzy inputs. Parallel computing technologies are used to speed up this process. The paper pro-poses an evolutionary algorithm that allows you to train a neuro-fuzzy system on a graphics processor unit. Including a model computational experiment.

Keywords

Mamdani type fuzzy systems, fuzzy truth value, evolutionary algorithm, graphics processor unit

1. Introduction

The paper considers fuzzy models of the Mamdani type [1] with many fuzzy inputs. The inference method generally has an exponential computational complexity. The proposed approach and the presented models are based on a fuzzy truth value and a measure of possibility. An evolutionary strategy (μ, λ) [2] is presented as a tuning algorithm. The implementation of this algorithm on a graphics processor is presented. The adequacy of the developed methods and algorithms has been proven during computational experiments.

The first section presents the statement of the problem and the estimation of the complexity of fuzzy inference. In the second section, the inference of the output value for the rule base is considered based on the decomposition theorem for a multidimensional membership function. The third section is about setting up and developing a machine learning algorithm. In the fourth section, an implementation of a training algorithm for a neuro-fuzzy system using parallel technologies is presented and a computational experiment is described.

2. Definition of the linguistic model

Define the linguistic model as a base of fuzzy rules R_k, $k = \overline{1,N}$:

$$R_k: If\ x_1\ is\ A_{1k} \# x_2\ is\ A_{2k} \# \ldots \# x_n\ is\ A_{nk}, then\ y\ is\ B_k\ with\ w_k, \quad (1)$$

where N is a number of fuzzy rules, $A_{ik} \subseteq X_i$, $i = \overline{1,n}$, $B_k \subseteq Y$ are fuzzy sets that are described by membership functions $\mu_{A_{ik}}(x_i)$ and $\mu_{B_k}(y)$ respectively. $x_1, x_2, \ldots, x_n$ are input variables of the linguistic model, and $[x_1, x_2, \ldots, x_n]^T = \boldsymbol{x} \in X_1 \times X_2 \ldots \times X_n$. Characters X_i, $i = \overline{1,N}$ and Y denoted respectively domain range of the input and output variables. In (1), linguistic bindings "AND" or "OR", denoted by «#».

In following notation $\boldsymbol{X} = X_1 \times X_2 \ldots \times X_n$ and $\boldsymbol{A_k} = A_{1k} \times A_{2k} \times \ldots \times A_{nk}$, the rule (1) represented as a fuzzy implication

$$R_k: \boldsymbol{A_k} \longrightarrow B_k,\ k = \overline{1,N}.$$

Russian Advances in Fuzzy Systems and Soft Computing: Selected Contributions to the 10th International Conference «Integrated Models and Soft Computing in Artificial Intelligence» (IMSC-2021), May 17–20, 2021, Kolomna, Russian Federation
EMAIL: vgsinuk@mail.ru (A. 1); panchenko.maks@gmail.com (A. 2)
ORCID: 0000-0002-0856-6637 (A. 1); 0000-0001-9032-2203 (A. 2)

R_k can be formalized as a fuzzy relation defined on a set $\boldsymbol{X} \times Y$, that is $R_k \subseteq \boldsymbol{X} \times Y$ is fuzzy set with membership function:

$$\mu_{R_k}(\boldsymbol{x}, y) = \mu_{\boldsymbol{A_k} \longrightarrow B_k}(\boldsymbol{x}, y).$$

The Mamdani model defines the function assignment like $\mu_{\boldsymbol{A_k} \longrightarrow B_k}(\boldsymbol{x}, y)$ based on known membership functions $\mu_{A_k}(x)$ and $\mu_{B_k}(y)$ in the following way [2, 6]

$$\mu_{\boldsymbol{A_k} \longrightarrow B_k}(\boldsymbol{x}, y) = T_1\left(\mu_{A_k}(x), \mu_{B_k}(y)\right) = \mu_{\boldsymbol{A_k}}(x) \overset{T_1}{*} \mu_{B_k}(y) \tag{2}$$

where $\overset{T_1}{*}$ is an arbitrary *t*-norm that is used as a parameter.

When training a system presented as (1), the fuzzy inference is $B'_k \subseteq Y$ if the inputs are fuzzy sets $\boldsymbol{A'} = A'_1 \times A'_2 \times \ldots \times A'_n \subseteq \boldsymbol{X}$ or $x_1 \; is \; A'_1 \; and \; x_2 \; is \; A'_2 \; and \; \ldots \; and \; x_n \; is \; A'_n$ with the corresponding membership function $\mu_{\boldsymbol{A'}}(x)$. In accordance with the generalized fuzzy modus ponens rule [2], fuzzy set B'_k determined by the composition of a fuzzy set $\boldsymbol{A'}$ and relation R_k, such

$$B'_k = \boldsymbol{A'} \circ (\boldsymbol{A_k} \to B_k)$$

or using the membership functions:

$$\mu_{B'_k}(y) = \sup_{x \in X} \{\mu_{\boldsymbol{A'}}(x) \overset{T_2}{*} (\mu_{\boldsymbol{A_k}}(x) \overset{T_1}{*} \mu_{B_k}(y))\} \tag{3}$$

where $\overset{T_2}{*}$ can be any *t*-norm. Complexity of the expression (3) is exponential $O(|X|^{\mathrm{n}} \cdot |Y|)$.

3. Inference method for the rule base with the center of sums defuzzification

To remove the exponential complexity problem of the (3), we use the following theorem. This theorem can be used under the condition that modeling the linguistic binding "AND" in the antecedent of rule (1) is used with t-norm "MIN", and for linguistic binding "OR" is used t-conorm "MAX".

Theorem

If $T(\mu_{A_{ik}}(x_i), \mu_B(y))$, $i = \overline{1,n}$ not increasing by argument $\mu_{A_{ik}}(x_i)$, then:

in case of linguistic binding "AND":

$$T\left(\mu_{A_k}(x), \mu_B(y)\right) = T\left(\min_{i=1,n}\{\mu_{A_{ik}}(x_i)\}, \mu_B(y)\right) = \min_{i=1,n}\left\{T\left(\mu_{A_{ik}}(x_i), \mu_B(y)\right)\right\},$$

in case of linguistic binding "OR":

$$T\left(\mu_{A_k}(x), \mu_B(y)\right) = T\left(\max_{i=1,n}\{\mu_{A_{ik}}(x_i)\}, \mu_B(y)\right) = \max_{i=1,n}\left\{T\left(\mu_{A_{ik}}(x_i), \mu_B(y)\right)\right\}.$$

Proof

Consider the proof of the first property.

Function $T(\mu_{A_{ik}}(x_i), \mu_B(y))$ is non-decreasing in argument $\mu_{A_{ik}}(x_i)$, if $\forall\, \mu_B(y) \in [0,1]$ from the condition $\mu_{A_{ik}}(x'_i) \le \mu_{A_{ik}}(x''_i)$ follows inequality:

$$T(\mu_{A_{ik}}(x'_i), \mu_B(y)) \ge T(\mu_{A_{ik}}(x''_i), \mu_B(y)) \tag{4}$$

This property is valid for any *t*-norm, according to their definition [6]

Assume any of the values $\mu_B(y) \in [0,1]$. Let also $(x_1, x_2, \ldots, x_n) \in X_1 \times X_2 \ldots \times X_n$ and define:

$$\mu_{A_{ik}}(x_e) = \min_{i=1,n}\{\mu_{A_{ik}}(x_i)\}. \tag{5}$$

This implies: $\mu_{A_{ik}}(x_e) \le \mu_{A_{ik}}(x_i) \;\; \forall i = \overline{1,n}$, in accordance with (4):

$$\mathrm{T}(\mu_{A_{ik}}(x_e), \mu_B(y)) \le \mathrm{T}(\mu_{A_{ik}}(x_i), \mu_B(y)) \;\; \forall i = \overline{1,n}\,, \tag{6}$$

then taking into account (5):

$$\mathrm{T}(\mu_{A_{ik}}(x_e), \mu_B(y)) = \underset{i=\overline{1,n}}{min}\{T(\mu_{A_{ik}}(x_i), \mu_B(y))\}. \quad (7)$$

Since $\mathrm{T}(\mu_{A_{ik}}(x_e), \mu_B(y))$ exists at the right part of the expression (7) and taking into account (5):

$$\mathrm{T}(\underset{i=\overline{1,n}}{min}\{\mu_{A_{ik}}(x_i)\}, \mu_B(y)) = \underset{i=\overline{1,n}}{min}\{T(\mu_{A_{ik}}(x_i), \mu_B(y))\}$$

The first property of the theorem is proved. The second property is proved similarly.

When the condition of the theorem on the decomposition of the multidimensional membership function (3) is fulfilled and when the linguistic binding "AND" is used in (1) , it will take the form:

$$\mu_{\mathrm{B}'_k}(y) = \min_{i=\overline{1,n}}\left\{\sup_{x_i\in X_i}\left\{\mu_{A'_i}(x_i) \overset{T_2}{*} T_1\left(\mu_{A_{ik}}(x_i),\ \mu_{B_k}(y)\right)\right\}\right\}, k = \overline{1,\mathrm{N}} \quad (8)$$

Note that expression (8) is characterized by complexity of the order $O(|v_i| \cdot |Y| \cdot n)$ i.e. corresponds to polynomial.

The membership function of one of the inputs can be defined as:

$$\mu_{A'_i}(x_i) = \tau_{A_{ik}/A'_i}(\mu_{A_{ik}}(x_i))$$

where $\tau_{A_k/A'}(\cdot)$ is a fuzzy truth value of a fuzzy set A_k in relation to A', representing the compatibility membership function $CP(A_k, A')$ A_k towards A', and A' is considered reliable [4]:

$$\tau_{A_{ik}/A'_i}(v_i) = \mu_{CP(A_{ik},A'_i)}(v_i) = \sup_{\substack{\mu_{A'_i}(x_i)=v_i \\ x_i\in X_i}}\left\{\mu_{A_{ik}}(x_i)\right\}, v_i \in [0,1].$$

Moving from variable *x* to variable *v*, denoting $\mu_{A_k}(x) = v$:

$$\mu_{A'_i}(x_i) = \tau_{A_{ik}/A'_i}(\mu_{A_{ik}}(x_i)) = \tau_{A_{ik}/A'_i}(v_i).$$

Then (8) can be represented through the fuzzy value of the degree of truth:

$$\mu_{\mathrm{B}'_k}(y) = \min_{i=\overline{1,n}}\left\{\sup_{v_i\in[0,1]}\left\{\tau_{A_{ik}/A'_i}(v_i) \overset{T_2}{*}\left(v_i \overset{T_1}{*}\ \mu_{B_k}(y)\right)\right\}\right\}, k = \overline{1,\mathrm{N}}. \quad (9)$$

If $T_1 = T_2 = T$, then considering the *t*-norm property of associativity, (9) can be converted to:

$$\begin{aligned}\mu_{\mathrm{B}'_k}(y) &= \min_{i=\overline{1,n}}\left\{\sup_{v_i\in[0,1]}\left\{\tau_{A_{ik}/A'_i}(v_i) \overset{T}{*}\left(v_i \overset{T}{*}\ \mu_{B_k}(y)\right)\right\}\right\} = \\ &= \min_{i=\overline{1,n}}\left\{\sup_{v_i\in[0,1]}\left\{(\tau_{A_{ik}/A'_i}(v_i) \overset{T}{*} v_i) \overset{T}{*}\ \mu_{B_k}(y)\right\}\right\} = \\ &= \min_{i=\overline{1,n}}\left\{\sup_{v_i\in[0,1]}\left\{\tau_{A_{ik}/A'_i}(v_i) \overset{T}{*} v_i\right\} \overset{T}{*}\ \mu_{B_k}(y)\right\} = \\ &\min_{i=\overline{1,n}}\left\{\Pi_{A_{ik}/A'_i} \overset{T}{*}\ \mu_{B_k}(y)\right\}, k = \overline{1,n},\end{aligned} \quad (10)$$

where

$$\Pi_{A_{ik}/A'_i} = \sup_{v_i\in[0,1]}\left\{\tau_{A_{ik}/A'_i}(v_i) \overset{T}{*} v_i\right\}$$

$\prod_{A_{ik}/A'_i}$ is a scalar value and a possibility measure of A_{ik} corresponds to the input A'_i or vice versa [5].

Taking into account the above transformations, we will get a crisp output value using the method of defuzzification by the center of sums [3]. In this case, the output value can be calculated as:

$$\bar{y} = \frac{\sum_{k=\overline{1,N}} \bar{y}_k . w_k \cdot \mu_{B'_k}(\bar{y}_k)}{\sum_{k=\overline{1,N}} w_k \cdot \mu_{B'_k}(\bar{y}_k)}, \quad (11)$$

where $\bar{y}$ is a crisp output value of system consisting of *N* rules (1); $\bar{y}_k$ are centers of membership functions $\mu_{B_k}(y)$, $k = \overline{1,\mathrm{N}}$:

$$\mu_{B_k}(\bar{y}_k) = \underset{y\in Y}{sup}\{\mu_{B_k}(y)\} = 1. \quad (12)$$

Since *t*-norms by definition satisfies the boundary condition $T(a;1) = a$, then substituting (10) into (11), considering (12), we will get:

$$\bar{y} = \frac{\sum_{\mathrm{K}=\overline{1,N}} \bar{y}_k . w_k . \underset{i=\overline{1,n}}{min}\left\{\Pi_{A_{ik}/A'_i}\right\}}{\sum_{\mathrm{K}=\overline{1,N}} w_k . \underset{i=\overline{1,n}}{min}\left\{\Pi_{A_{ik}/A'_i}\right\}}. \quad (13)$$

4. Configuration of fuzzy model and learning algorithm

We define the training and test sample, respectively, as a set of pairs of input and output values

$$(X^r, y^r), r = \overline{1, M},$$

where $X^r = (x_1^r, x_2^r, \dots, x_n^r)$ is an input vector in r-pair, y^r is a corresponding output.

The input values can be crisp values, which are then fuzzy using fuzzification before output, or fuzzy values specified by terms, for example, "low", "medium", "high", etc., which are defined in the knowledge database. When using fuzzy values, the inputs will be unchanged throughout the training; when using numerical values opposite, the inputs can change during training when changing the parameters that define the terms of the membership functions.

Tuning a fuzzy model consists in finding its parameters that minimize deviations between the desired and actual behavior of the model. It is assumed that the desired behavior of the model is given by a fuzzy training sample.

Denote $\mu_{A_{ij}}(x_i)$ the function of membership of the input to the fuzzy term A_{ij}, where $i, j = \overline{1, n}$. The membership function is defined on the domain of the linguistic variable.

An example of table styling. It is recommended to add cross references to tables, i.e., please, check Table 1. The style should be switched to Normal.

The Gaussian function will be used to describe the membership function:

$$\mu_T(x) = e^{-\frac{(x-b)^2}{2c^2}}, \quad (14)$$

where b and c are the coordinate of the maximum and the concentration coefficient of the membership function of the fuzzy set T.

In the implemented inference method, it is possible to use different t-norms. During training, it is also possible to select the most suitable t-norm.

The parameters b and c of the membership functions of all terms of the system, the weighting coefficients of the rules, as well as the type and parameter of the t-norm will be adjusted during the training of the system.

Let's define a fuzzy model of the object $y = f(x_1, x_2, \dots, x_n)$ in the following form:

$$y = F(X, B, C, W, t, \gamma),$$

where $X = (x_1, x_2, \dots, x_n)$ is an input vector; $B = (b_1, b_2, \dots, b_q)$ and $C = (c_1, c_2, \dots, c_q)$ are vectors of the membership functions parameters (14) of fuzzy terms from the knowledge base; $W = (w_1, w_2, \dots, w_N)$ is the vector of weighting coefficients of fuzzy rules (1);

N – the total number of rules (1);

q – total number of terms;

t, γ – type and parameter of t-norm;

F – the Input / Output communication function corresponding to the (13).

The setting of the fuzzy model is formulated as follows optimization problem [2]: find a vector (B, C, W, t, γ)

$$R = \sqrt{\frac{1}{M}\sum_{r=\overline{1,M}}[y^r - F(X^r, B, C, W, t, \gamma)]^2} \to min. \quad (15)$$

An evolutionary strategy (μ, λ) [2] is used for training, since when it is used, the probability that the local minimum from the search for solutions will be issued as a solution of the problem is reduced.

In this case, an individual is a genotype that consists of two chromosomes. Chromosome x contains N genes-parameters of the system, and chromosome σ contains the values of standard deviations used in the process of mutation of the corresponding genes of chromosome x. The tuning algorithm can be described in the following sequence of steps:

1. the parental population *P* is formed, which contains μ individuals with chromosome *x*, containing the system parameters set by the expert, the genes of the σ chromosome in all individuals are equal to 1;
2. reproduction is performed to form a temporary population *T* of size λ, with $\lambda > \mu$, from the parental population *P*;
3. a population of seeds *O* is formed by mutation of the temporary population *T*;
4. the population *O* is estimated. To estimate the individual, a system is constructed with the parameters recorded in the *x* chromosome. Further, for each pair of *"inputs-output"* from the training

sample, the square of the difference between the output of the system, which receives the input, and the "output" is calculated. The sum of the obtained squares is the estimate of the given individual;

5. a new parental population *P* is formed from μ of the best individuals of the population of seeds *O*;
6. the condition of the completion of the algorithm is checked and if it is not met, then it is necessary to repeat steps 2-6 with the new parent population *P*;
7. the best individual is displayed.

Let us consider some of the steps of the algorithm. When the first parental individual is generated, the σ chromosome is filled with single values and then changes them only by mutation. All individuals of the population undergo mutation operations, and each separately. Both chromosomes mutate, but the mutation process is different. The chromosome $\sigma = [\sigma_1^2, \dots, \sigma_n^2]^T$ mutates first according to the expression:

$$\sigma_i' = \sigma_i \exp\left(\tau' N(0,1) + \tau N_i(0,1)\right),$$

where i = 1, ..., n, n is the length of the chromosome; N (0,1) - a random number with a normal distribution (generated once for the entire chromosome); N_i (0,1) - random number with normal distribution (generated for each gene separately); τ' and τ are parameters of the evolutionary strategy that affect the convergence of the algorithm. In the literature, there is a form where *C* most often equal to *1*:

$$\tau' = \frac{C}{\sqrt{2n}}, \qquad \tau = \frac{C}{\sqrt{2\sqrt{n}}}.$$

New values of the mutation σ_i' affect the change in the value of x_i, as follows:

$$x_i' = x_i + \sigma_i' N_i(0,1), \tag{16}$$

where $N_i(0,1)$ is a normally distributed random number, $i = 1, \dots, n$ [2].

The operation of mutation of the chosen evolutionary strategy was improved by adding restrictions on the values that the genes of the *x* chromosome can take. For each gene of the chromosome, before the start of the algorithm, the minimum and maximum possible values are selected. If during the mutation the value of a gene or allele goes beyond the specified range of possible values, the new value is calculated as:

$$x_i'' = \begin{cases} a_{max} - c, x_i' > a_{max}, \\ a_{min} + c, x_i' < a_{min}, \end{cases}$$

where x_i is the value of the gene obtained by the mutation operation; a_{min} and a_{max} are the minimum and maximum possible values of the gene, respectively; *c* is a correction value equal to

$$c = |x_i' - [x_i'/(a_{\max} - a_{min})]|.$$

After changing the property of the gene, the corresponding gene of chromosome σ should also be changed using (16):

$$\sigma_i'' = (x_i'' - x_i)/N_i(0,1).$$

Thus, as a result of the mutation operation, the values of the genes are x_i'' and σ_i''.

Applying this algorithm for training a fuzzy system, function (15) acts as an objective function. The best is an individual whose objective function value is less than that of other individuals in the population.

The condition for the completion of the algorithm can be the number of individuals or the achievement of a specified estimate by an individual of the population *P*.

5. Implementation of the learning algorithm on a graphic processor and computing experiment

Learning operations were carried out on a graphics processor using OpenCL technology [6].

The population consists of a number of individuals. In computer memory, the population is represented as an array, the element of which is an individual.

For the reproduction operation, a temporary population was generated in the form of an array of size λ, containing numbers from *0* to *(μ-1)* (μ is the size of the parental population). This array defines the indices of individuals that will be included in the new temporary population *T*.

The mutation operation is performed for each individual of the temporary population separately. The mutation algorithm of the evolutionary strategy provides for the use of normally distributed random

numbers. This is an expensive operation on the GPU and requires the creation of a random number generator, so an array of random numbers of size $\lambda * 2 * n$ is pre-generated.

A block diagram of the mutation operation for each individual is shown in Fig. 1, where *orgs* is an array of individuals in the population, *i* is the index of an individual undergoing mutation, *t* and *t1* are parameters of the evolutionary strategy, *rands* is an array of random numbers, *limitations* is an array of restrictions for the values of genes, *mutated_orgs* is an array of modified individuals.

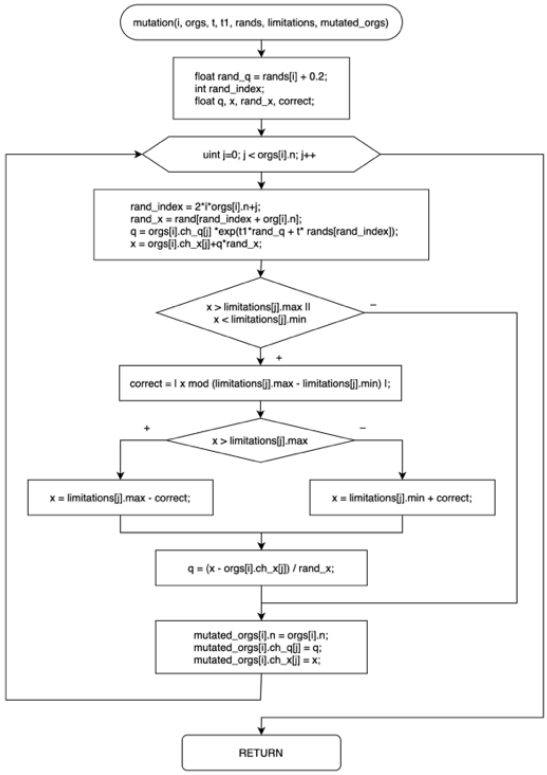

Figure 1: Mutation flow chart

The modified individuals are estimated according to (15), the output of the fuzzy system is also calculated on the graphics processor. This process is described in [7].

The selection operation is organized by sorting the individuals in ascending order of the obtained estimates and the subsequent selection of the first μ individuals. If the condition for the completion of the algorithm is the achievement of a certain average score, then the average score is calculated for the selected individuals.

To implement a computational experiment, we consider an object with two inputs $x_1, x_2 \in [0, 10]$ and one output *y* given by the relation:

$$y = (x_1 - 7)^2 \cdot \sin(0.61 \cdot x_2 - 5.4). \quad (17)$$

The task was set according to the graph of the reference dependence shown in Fig. 2, create a fuzzy model and tuning it using a fuzzy training sample. The adequacy of the fuzzy model must be checked by criterion (15) on a test sample of randomly generated input-output pairs.

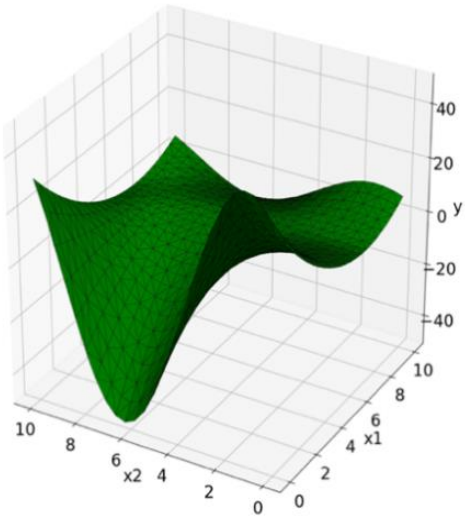

Figure 2: Model of nonlinear dependency (17)

The fuzzy knowledge base was visually generated by an expert based on Fig. 2. It consists of seven rules, which are summarized in table. 1 For the linguistic assessment of the input variables x_1 and x_2 and the output variable y, the terms *"Low" (L)*, *"Below Average" (BA)*, *"Medium" (M)*, *"Above Average" (AA)* and *"High" (H)*. In this case, the value of criterion (15) on the test sample is 11.25 (Fig. 3a). The weight coefficients (*W*) of the rules of these fuzzy models are shown in Table 1. Testing of the trained fuzzy models (Fig. 3b) indicates an acceptable quality of identification of the nonlinear dependence (17).

To study the process of tuning a fuzzy model, training samples of 10, 20, …, 100 pairs of "inputs-outputs" were used. The training sample was generated as randomly selected input values from the domain of their definition and the output values calculated according to (17), which were then used to evaluate the identification accuracy criterion. To turn this training sample into a fuzzy one, the generated clear values of the input variables were estimated in terms of linguistic variables before the model was tuning, the output values remained unchanged (crisp).

Table 1
Fuzzy knowledge base

x_1	x_2	y	W	
			before tuning	after tuning
low	low	high	1	0.61
low	average	low	1	0.63
low	high	high	1	0.20
average	—	average	1	0.99
high	low	above average	1	0.10
high	average	below average	1	0.21
high	high	above average	1	0.01

Fig. 4 shows the learning curves of fuzzy models. They reflect the dependence of identification errors (15) on the size (*M*) of the training sample itself on the training and test samples. The tuning was carried out using an evolutionary strategy $(\mu, \lambda), \mu = 40, \lambda = 160$ over 100 iterations. Each point on the learning curves (Fig. 4) was calculated as the average of the experimental results for 10 different training samples. With an increase in the size of the fuzzy training sample, the residual on the test sample decreases, and the difference between the residuals on the training and test sample decreases.

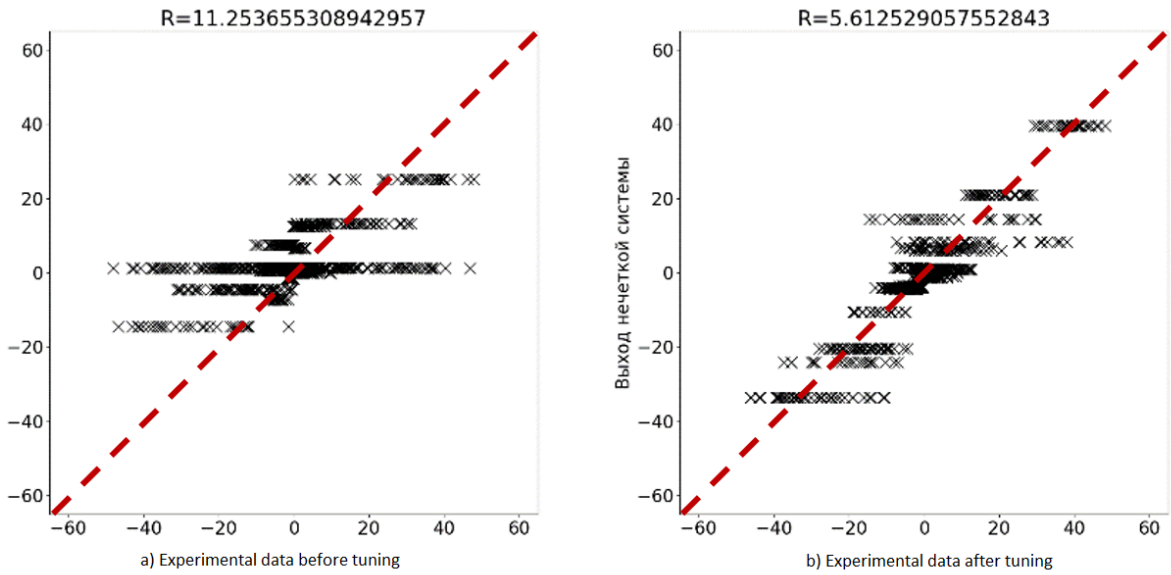

Figure 3: Test of the fuzzy system

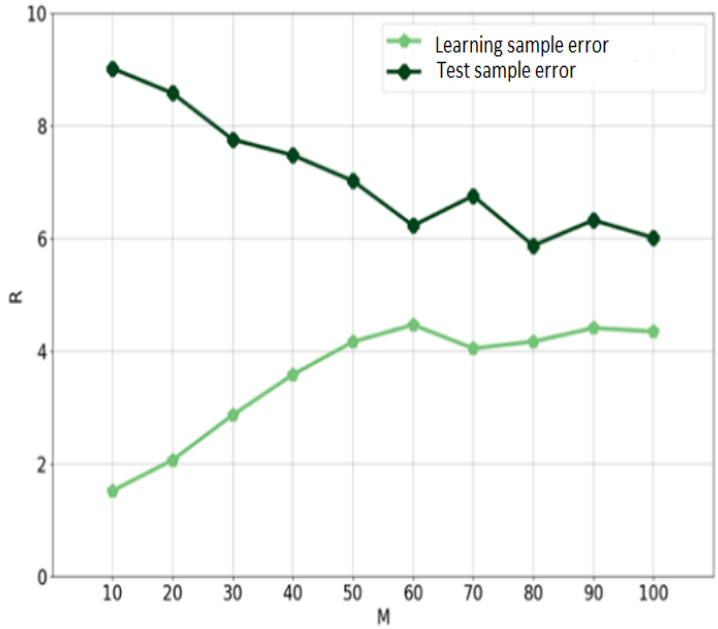

Figure 4: Learning curves

6. Conclusion

The article presents an inference method for Mamdani-type systems with fuzzy inputs with polynomial computational complexity. An algorithm for training such systems using a fuzzy training sample and using parallel technologies has been developed. The conducted computer experiments show that the fuzziness in the experimental data reduces the training efficiency, but with an increase in the training sample, the training accuracy increases.

7. Acknowledgements

This work was supported by grants from the Russian Foundation for Basic Research, project 20-07-00030 A and project 19-29-09056 mk.

8. References

[1] Yu. I. Kudinov, A. Y. Kelina, I. Y. Kudinov, Fuzzy models and control systems, Editorial URSS, Moscow, 2017.

[2] L. Rutkovsky, Methods and technologies of artificial intelligence, Goryachaya liniya – Telekom, Moscow, 2010.

[3] A. V. Andreychikov, O. N. Andreychikova, Science and decision making, Book 2: Decision making in conditions of uncertainty, Lenand, Moscow, 2021.

[4] V. G. Sinyuk, V. V. Mikhelev, Methods of inference for logical-type systems based on the fuzzy degree of truth, in: Proceedings of the Russian Academy of Sciences, Theory and control systems, 2018, pp.108-115.

[5] Yu. A. Zak, Decision making in the conditions of fuzzy and blurry data: Fuzzy-technologies, Lenand, Moscow, 2016.

[6] D. L. Chopp, Introduction to High Performance Scientific Computing, 1st. ed., SIAM, University City, Philadelphia, 2019.

[7] V. G. Sinyuk, V. M. Polyakov, M. V. Panchenko, Output method for MISO-structure systems with fuzzy inputs using parallel computing technologies, VIII, volume 1 of International Scientific and Practical Conference, Fuzzy Systems, Soft Computing and Intelligent Technologies (FSSCIT-2020), Universum, Smolensk, 2020, pp. 54-64.