Information Sciences 181 (2011) 1591–1608

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

# Type-2 fuzzy neural networks with fuzzy clustering and differential evolution optimization

Rafik A. Aliev [a,*], Witold Pedrycz [b], Babek G. Guirimov [a], Rashad R. Aliev [c], Umit Ilhan [c], Mustafa Babagil [c], Sadik Mammadli [d]

[a] *Azerbaijan State Oil Academy, 20 Azadlyg Ave., Baku, Azerbaijan*
[b] *University of Alberta, Canada*
[c] *Eastern Mediterranean University, Cyprus*
[d] *Azerbaijan State Oil Academy, Azerbaijan*

A R T I C L E  I N F O

*Article history:*
Received 12 February 2010
Received in revised form 9 October 2010
Accepted 27 December 2010
Available online 1 January 2011

*Keywords:*
Type-2 fuzzy neural network
Fuzzy clustering
Type-2 fuzzy rule base
Differential evolution optimization

A B S T R A C T

In many real-world problems involving pattern recognition, system identification and modeling, control, decision making, and forecasting of time-series, available data are quite often of uncertain nature. An interesting alternative is to employ type-2 fuzzy sets, which augment fuzzy models with expressive power to develop models, which efficiently capture the factor of uncertainty. The three-dimensional membership functions of type-2 fuzzy sets offer additional degrees of freedom that make it possible to directly and more effectively account for model's uncertainties. Type-2 fuzzy logic systems developed with the aid of evolutionary optimization forms a useful modeling tool subsequently resulting in a collection of efficient "If-Then" rules.

The type-2 fuzzy neural networks take advantage of capabilities of fuzzy clustering by generating type-2 fuzzy rule base, resulting in a small number of rules and then optimizing membership functions of type-2 fuzzy sets present in the antecedent and consequent parts of the rules. The clustering itself is realized with the aid of differential evolution.

Several examples, including a benchmark problem of identification of nonlinear system, are considered. The reported comparative analysis of experimental results is used to quantify the performance of the developed networks.



## 1. Introduction

During the past decades, intelligent technologies have been emerged as an interesting and relevant alternative to solve many real-world problems in forecasting, decision making, control, and decision analysis. Among these approaches, Soft Computing encompassing fuzzy logic, neural networks, and evolutionary optimization methods, has been viewed as an interesting and useful alternative [2]. Fuzzy logic systems (i.e., type-1 fuzzy logic systems) have been successfully used in various applications.

To design of "ordinary" fuzzy logic systems, knowledge of human experts along with available data are utilized for the construction of fuzzy rules and membership functions based on available linguistic or numeric information [28]. However, in many cases available information or data are associated with various forms of uncertainty which should be taken into account. The uncertainty can be captured by using higher order and/or higher type fuzzy sets. In this regard, type-2 fuzzy

* Corresponding author. Tel./fax: +994 12 5984509.
*E-mail addresses:* raliev@asoa.edu.az (R.A. Aliev), pedrycz@ee.ualberta.ca (W. Pedrycz), b.guirimov@azbtc.com (B.G. Guirimov), rashad.aliyev@emu.edu.tr (R.R. Aliev).

0020-0255/$ - see front matter 
doi:10.1016/j.ins.2010.12.014

sets have been considered as a viable conceptual and algorithmic vehicle to design intelligent systems. In many cases reported in the literature, it was demonstrated that type-2 fuzzy sets contribute to the robustness and stability of the resulting constructs (namely, models, inference schemes, classifiers, etc.) [6,14–20].

The concept and properties of type-2 fuzzy sets were originally introduced by Zadeh [38] and were further pursued by others, cf. [16–19,21,24]. Recently type-2 fuzzy sets have been applied to several areas including control of mobile robots [22,36], decision making [6,37], forecasting of time-series [20], identification of nonlinear plants [34], diagnosis [14], scheduling [15], and others.

In this applied context, one should make a remark that the use of type-2 fuzzy logic systems usually increases the computational complexity in comparison with the level of computational overhead encountered in type-1 systems due to the use and optimization of three-dimensional membership functions describing type-2 fuzzy sets [23]. Given that the use of type-2 fuzzy logic system could provide a significant level of improvement of performance, it is worth accepting the increased computational complexity associated with their usage [13].

In [13] authors focused on interval type-2 fuzzy clustering problems by studying an extension of the commonly known Fuzzy C-Means (FCM algorithm). In this research, the uncertainty becomes associated with various values of the fuzzifier parameter $m$ that controls the amount of fuzziness present in the final partition, see also [33]. In [4] the author suggests a method for accelerating fuzzy clustering by using gradient-based neural network training algorithms. Authors of [9,32] apply evolutionary algorithms for generation of clusters. In [27] discussed was an important question of determination of effective upper and lower bounds of the fuzzifier parameter $m$.

Some studies appeared recently on type-2 fuzzy inference systems [11] or, being more general, on type-2 fuzzy neural networks and their efficient training algorithms [5,25]. Very limited research has been realized on applications of type-2 fuzzy sets in the area of fuzzy rule extraction. In [34] type-2 fuzzy logic system cascaded with neural network is presented to handle uncertainty with dynamical optimal learning. The considered type-2 fuzzy neural system consists of type-2 fuzzy sets forming the antecedent part while a two-layer interval neural network is used to form the consequent part. One could agree with the authors of [10] that from the conceptual standpoint, one can view the publication [34], in spite of some of its mathematical flaws, as the first study on type-2 fuzzy neural network aimed at uncertainty handling.

In spite of intensive developments of the theory and design methods for type-2 fuzzy logic systems [5,11,12,16,17,19,24,25], there has been a fairly limited progress in the area of type-2 fuzzy rule extraction, tuning of parameters of membership functions used in the rules, merging type-2 fuzzy logic system with other components of Soft Computing, namely, with neural networks, evolutionary computing, and type-2 inference schemes. These are visible shortcomings which have suppressed pursuits in the realm of type-2 fuzzy modeling.

Bearing this in mind, our objective is to develop a category of type-2 fuzzy neural networks based on the constructs of type-2 fuzzy sets with these fuzzy sets forming a backbone of a collection of "If-Then". The underlying construction of information granules (type-2 fuzzy sets) as well as the learning mechanisms for the developed Type-2 Fuzzy Inference Neural Network (T2FINN) are realized by means of differential evolution (DE) [31].

The study is structured as follows. In Section 2, we cover all prerequisite material (such as a brief summary on type-2 fuzzy sets, type-2 fuzzy logic system, type-2 neural network, and differential evolution) to be used in the study. In Section 3, we formulate the problem in detail. In Section 4, we consider type-2 fuzzy rule extraction using type-2 fuzzy clustering. Furthermore a formation of initial membership functions is discussed. In Section 5, Type-2 Fuzzy Inference Neural Network (T2FINN) structure and inference logic is presented and the optimization of rule base and tuning parameters of membership functions by DE are discussed. In Sections 6 and 7, we present computer simulations and elaborate on applications of the proposed approach to function approximation, nonlinear plant identification, cement and oil products time-series forecasting, and loan assessment problems. Concluding comments are included in Section 7.

## 2. Preliminaries

In this section, we briefly introduce the main prerequisites that are helpful in further detailed investigations carried out in this study.

### 2.1. Type-2 fuzzy sets [16,21,23]

A type-2 fuzzy set $\widetilde{A}$ is characterized by type-2 membership function $\mu_{\widetilde{A}}(x,h)$ where $x \in X$ and $h \in H_x \subseteq [0,1]$. Formally, $\widetilde{A}$ can be expressed as:

$$\widetilde{A} = \left\{((x,h), \mu_{\widetilde{A}}(x,h)) | \forall x \in X, \forall h \in H_x \subseteq [0,1]\right\}, \tag{1}$$

where $\mu_{\widetilde{A}}(x,h) \leqslant 1$. $H_x \subseteq [0,1]$ is called a primary membership of $x$. Uncertainty in the primary membership of type-2 fuzzy set $\widetilde{A}$ is represented by of a bounded region (called a footprint of uncertainty (FOU), which is determined as:

$$FOU(\widetilde{A}) = \bigcup_{x \in X} H_x. \tag{2}$$

Several examples of type-2 membership functions and FOUs are given in Fig. 1.

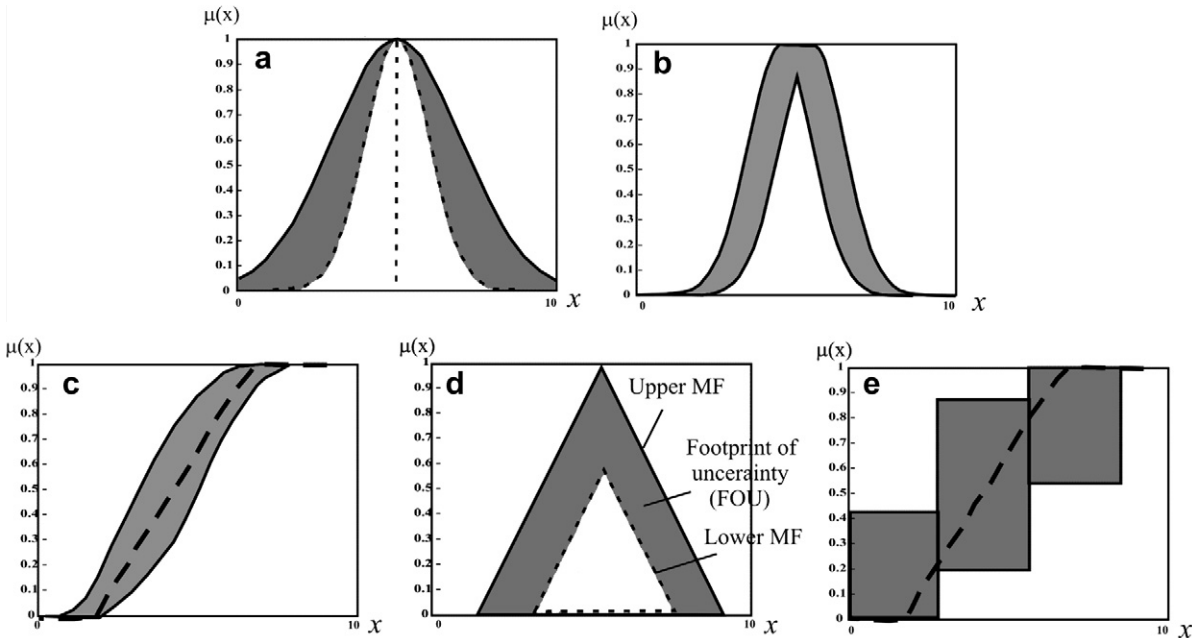


**Fig. 1.** Examples of membership functions (MFs) and associated FOUs: (a) Gaussian MF with uncertain spread; (b) Gaussian MF with uncertain modal value; (c) sigmoid MF with inflection uncertainty; (d) triangular type-2 MF; (e) Granulated sigmoid MF with granulation uncertainties.

The shaded FOUs imply that the resulting construct is three-dimensional meaning that there is a certain distribution of the membership grades located in this region. A type of these distributions depends on a way in which such secondary grades are assigned and what numeric values they assume. If all of these grades are equal to one, then the resulting type-2 fuzzy sets are called interval type-2 fuzzy sets and can be regarded as a specific case of type-2 fuzzy sets.

### 2.2. Type-2 fuzzy logic system (FLS)

In general, a type-2 FLS consists of four functional components: a fuzzifier or coder (realizing a mapping input data into type-2 fuzzy sets), type-2 fuzzy rule base, inference engine, and output processor (decoder). The fuzzy rule base is expressed as a collection of "If-Then" statements, whose antecedents and consequents are type-2 fuzzy sets. The i-th fuzzy rule with $s$ antecedents is expressed as:

$$R^i : \text{If } x_1 \text{ is } \widetilde{A}_1^i \text{ and } x_2 \text{ is } \widetilde{A}_2^i \text{ and} \ldots x_s \text{ is } \widetilde{A}_s^i \text{ Then } y \text{ is } \widetilde{B}^i, \tag{3}$$

where $x_j$ ($j = 1,2,\ldots,s$) and $y$ are input and output variables, respectively; $A_j^i (j = 1, 2, \ldots, s)$ and $\widetilde{B}^i$ are antecedent and consequent type-2 fuzzy sets, respectively.

Quite commonly, the inference engine of type-2 fuzzy FLS uses the extended sup-star composition. The input type-2 fuzzy sets occurring in the antecedent part of a rule are connected by some meet operation whereas the output type-2 fuzzy sets are combined using joint operation, see [21,26].

The output processor of type-2 FLS consists of a two-phase defuzzification (decoding). Type reducer converts type-2 fuzzy output sets to type-1 output fuzzy sets. An "ordinary" defuzzifier realizes further conversion of type-2 fuzzy sets into a single numeric value and uses one of the well-known methods commonly encountered in a type-1 FLS.

### 2.3. DE optimization method

Recently, many heuristic algorithms have been proposed for global optimization of nonlinear, non-convex, and non-differentiable functions [1,7,29,35]. These methods are more flexible than classical ones as they do not require differentiability, continuity, or other restrictive properties which are usually required for the objective function to be optimized. Some of such methods are genetic algorithms, evolutionary strategies, particle swarm optimization, and differential evolution (DE). In this study, we consider the use of the DE algorithm. This population based algorithm implements global search. Being designed specifically for numerical optimization, it is characterized by good convergence properties in multidimensional search spaces. These characteristics of DE make the method an efficient tool for implementation of neural network training.

As being stochastic in its nature, DE algorithm uses an initial population of randomly generated individuals and applies to them operations of differential mutation, crossover, and selection [31]. DE considers individuals as vectors in $n$-dimensional

Euclidean space. The population of *ps* individuals is maintained through consecutive generations. A new vector is generated by mutation, which, in this case is completed by adding a weighted difference vector of two individuals to a third individual as follows: $\mathbf{S}_{new} = (\mathbf{S}_{r1} - \mathbf{S}_{r2})f + \mathbf{S}_{r3}$, where $\mathbf{S}_{r1}$, $\mathbf{S}_{r2}$, $\mathbf{S}_{r3}$ ($r_1 \neq r_2 \neq r_3$) are 3 different individuals randomly picked from the population and $f$ (>0) is the mutation parameter. The mutated vector then undergoes crossover with another vector thus generating a new offspring.

The selection process is realized as follows. If the resulting vector yields a lower value of the objective function than the member of the population with an index changing consequently, the newly generated vector will replace the vector with which it was compared in the following generation. Another approach, which we adopted in this research, is to randomly pick an existing vector for realizing crossover.

Extracting distance and direction-like information from the population to generate random deviations results in an adaptive scheme with excellent convergence properties. DE has been successfully applied to solve a wide range of problems such as those found in image classification, clustering, optimization, etc.

Fig. 2 illustrates a process of generation of a new trial solution (vector) $\mathbf{S}_{new}$ from three randomly selected members of the population $\mathbf{S}_{r1}$, $\mathbf{S}_{r2}$, $\mathbf{S}_{r3}$. Vector $\mathbf{S}_i$, $i = 1,\ldots,ps$, becomes the candidate for replacement by the new vector, if the former is better in terms of the DE cost function. Here, for illustrative purposes, we assume that the solution vectors are of dimension $n = 2$ (i.e., 2 parameters to be optimized).

### 2.4. Type-2 fuzzy neural networks

A type-2 fuzzy neural network implements type-2 fuzzy logic system and some of their parameters and components are presented by fuzzy logic terms [34]. Successive layers of the network perform type-2 fuzzification, represent fuzzy rules, define the consequences of each rule, and realize an aggregation of the type-2 fuzzy (or type-1 fuzzy) output values for each output variable. Finally, the defuzzification takes place. The T2FINN of this topology can effectively represent FLS with type-2 fuzzy "If-Then" rules.

Another type of a type-2 fuzzy neural network adapts the structure and functionality of "traditional" feedforward or recurrent neural networks exhibiting several layers of processing neurons of the same type. In this case, numeric weights and thresholds (and recurrent weights in the case of a recurrent NN) become type-2 fuzzy numbers (sets). Note that in this case all processing in neurons is based on type-2 fuzzy arithmetic. Inputs to the network and outputs from the network are also type-2 fuzzy values. Such networks can also implement type-2 fuzzy "If-Then" rules representation and inference. However these networks need more intensive processing.

Authors of [12] analyzed approximation capabilities of multilayer fuzzy neural networks for the set of fuzzy-valued functions.

## 3. Problem statement

Our aim is to develop a design methodology and discuss ensuing algorithmic aspects arising within the realm of designing of type-2 fuzzy neural networks. More specifically, we.

(1) generate type-2 fuzzy "If-Then" rule base from data by making use of clustering augmented by the mechanisms of evolutionary optimization (in this study, we confine ourselves to the optimization framework of differential evolution), and
(2) adjust (optimize) the parameters of the initial rule-base on the basis of a type-2 fuzzy neural inference system using the DE optimization (DEO, for short).

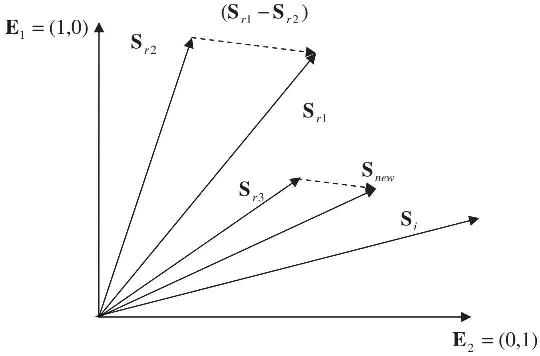


**Fig. 2.** Realization of DE optimization: a two-dimensional case.

Given $n$ input–output pairs of data $\{\mathbf{X}_1/Y_1, \mathbf{X}_2/Y_2, \ldots, \mathbf{X}_n/Y_n\}$ and the required accuracy of the model ($\varepsilon \geqslant 0$), we form the minimal number of the rules and parameters of the type-2 membership functions (for instance, 6 parameters describing each interval valued triangular membership function described here as $T(LL, LR, ML, MR, RL, RR)$ for input and output terms) so that the error function denoted as $Err = \sum_{i=\overline{1,n}} \|y(\mathbf{X}_i) - Y_i\|$ (where $y(\mathbf{X})$ is the inference system's numeric output for any given input data vector $\mathbf{X}$ of dimension $s$: $\mathbf{X} = [x_1 x_2 \ldots x_s]^{\mathrm{T}}$) satisfies the inequality $Err < \varepsilon$. The fuzzy model output will be obtained on the basis of the inferencing from the following "If-Then" rules:

$$R^i: \text{If } x_1 \text{ is } \widetilde{A}_1^i \text{ and } x_2 \text{ is } \widetilde{A}_2^i \quad \text{and} \quad \ldots x_s \text{ is } \widetilde{A}_s^i \text{ Then } y \text{ is } \widetilde{B}^i, \tag{4}$$

where $x_j$ $(j = \overline{1,s})$ and $y$ are input and output variables, respectively; $\widetilde{A}_j^i = 1, 2, \ldots, s)$ and $\widetilde{B}^i$ are antecedent and consequent type-2 fuzzy sets, respectively.

## 4. Type-2 fuzzy clustering and rule extraction

Fuzzy clustering is a well-established paradigm used to generate the initial type-2 fuzzy "If-Then" model. When dealing with fuzzy clustering, cf. [13], there are several essential parameters whose values need to be decided upon in advance. We can think of uncertainty, which is inherently associated with the selection of the specific numeric values of these parameters. In the FCM-like family of fuzzy clustering, the fuzzification coefficient (fuzzifier) plays a visible role as it directly translates into the shape (geometry) of resulting fuzzy clusters. Let us recall that for the values of "$m$" close to 1, the membership functions become very close to the characteristic functions of sets whereas higher values of "$m$" (say, over 3 or 4) result in "spiky" membership functions.

The type-2 fuzzy clustering may be formulated as follows (for simplicity, we consider here interval fuzzy clustering). $n$ data vectors $P = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n\}$ are given. We partition the data $P$ into $c$ fuzzy clusters so that the following objective function is minimized:

$$J_{\tilde{m}} = \sum_{i=1}^{n} \sum_{j=1}^{c} \tilde{u}_{ij}^{\tilde{m}} \|\mathbf{p}_i - \tilde{\mathbf{v}}_j\|^2, \tag{5}$$

where $\|*\|$ is Euclidean distance.

The minimization of (5) is carried out under the two "standard" requirements commonly imposed on the partition matrix: (1) the clusters are non-empty and there are more than a single cluster; (2) the sum of membership degrees of any data point is equal to 1.

Here the fuzzifier $\tilde{m}$ represents interval (type-1 fuzzy) value ($[m_1, m_2]$); $\tilde{\mathbf{v}}_j$ is the prototype of the $j$th cluster generated by fuzzy clustering; $\tilde{u}_{ij}$ is the membership degree of the $i$-th data belonging to the $j$th cluster represented by the prototype $\tilde{\mathbf{v}}_j$.

The choice of the number of clusters can be realized on a basis of some cluster validity criterion. The well-known validity indexes PC (partition coefficient) and PE (partition entropy) suggested by Bezdek has some shortcomings, main of which is monotonic dependence on the number of clusters ($c$) [3,8]. Here, we adopt the following validity measure [8] for type-2 fuzzy clustering, which overcomes the main disadvantage of PC and PE:

$$V(u, c) = \frac{1}{n} \sum_{k=1}^{n} \max(u_{ij}) - \frac{1}{K} \sum_{i=1}^{c-1} \sum_{\substack{j=i+1 \\ i}}^{c} \left[ \frac{1}{n} \sum_{k=1}^{n} \min(u_{ik}, u_{jk}) \right], \tag{6}$$

where $K = \sum_{i=1}^{c-1} i$.

This validity index is chosen as it takes into account both compactness (the first term) and separation of clusters in its partitioning (the second term). Combining the information on fuzzy compactness and separation, this validity index tends to indicate a good cohesion within clusters and small overlap between pairs of clusters [8].

The number of clusters can be defined on the basis of the maximal value of $V$, which is picked up for different values of $\tilde{m}$. The minimum number of clusters and, therefore, the number of the fuzzy rules themselves is very important for interpretability reasons. However, this number may need to be increased based on the required accuracy (expressed via the MSE performance index).

Clustering is completed through the minimization of the values of the objective function $J_{\tilde{m}}$ with the fuzzifier being considered as a fuzzy (interval) number $\tilde{m} = [m_1, m_2]$. The most widely used clustering is that of the Fuzzy C-Means (FCM) [6,8,13] due to its efficiency and simplicity. Standard FCM is an iterative procedure that leads to a local minimum of the objective functions (5).

Based on the experimental work reported in [27] we reformulate the problem (5) as:

$$\begin{aligned} J_{m1} &= \sum_{i=1}^{n} \sum_{j=1}^{c} u_{ij}^{m_1} \left\| \mathbf{p}_i - \mathbf{v}_j^{(1)} \right\| \to \min, \\ J_{m2} &= \sum_{i=1}^{n} \sum_{j=1}^{c} u_{ij}^{m_2} \left\| \mathbf{p}_i - \mathbf{v}_j^{(2)} \right\| \to \min, \end{aligned} \tag{7}$$

subject to constraints:

$$0 < \sum_{i=1}^{n} u_{ij} < n \ (j = 1, 2, \ldots, c) \quad \text{and} \quad \sum_{j=1}^{c} u_{ij} = 1 \ (i = 1, 2, \ldots, n).$$

The vector $\tilde{\mathbf{v}}_i$ is formed as:

$$\tilde{\mathbf{v}}_i = \left[\min\left(\mathbf{v}_i^{(1)}, \mathbf{v}_{Ind_i}^{(2)}\right), \max\left(\mathbf{v}_i^{(1)}, \mathbf{v}_{Ind_i}^{(2)}\right)\right], \tag{8}$$

where $Ind_i = \arg\min_j \left\|\mathbf{v}_i^{(1)}, \mathbf{v}_j^{(2)}\right\|$.

The conclusion is based on the result that the change in the location of the prototypes associated with a change of the values of $m$ is of monotonic character. The above assumption allows us to replace (5) by (7) and significantly reduce computational burden when searching for cluster centers using different optimization methods. As shown in [27] the meaningful range for $m$ is [1.4, 2.6].

In [13], the authors presented a type-2 fuzzy version of the FCM considering interval type-2 fuzzy fuzzifiers. However, gradient-based optimization methods (such as the FCM itself) exhibit here some disadvantages. One of significant drawbacks is that they may not produce a global minimum but instead could get stuck in some local minimum. On the other hand, the standard iterative scheme may not be applicable directly to the considered problem, especially when various distance functions other than the Euclidean one are being used. As pointed out in [29], a better approach could be to consider a population-based algorithm such as genetic algorithm, DE, or PSO [30]. In this work we use DEO – a simple, yet powerful global optimizer.

An illustrative example of the clustering problem is shown in Fig. 3. There is a collection of two-dimensional data with clearly visible four clusters. The data points are separated into 4 clusters. Fig. 3(a) shows the clusters found by the standard FCM with $m = 2$. Fig. 3(b) shows type-2 fuzzy clustering result for the same problem. In Fig. 3b the squares around the cluster centers, shown by cross-signs ($\times$), incorporate the cluster centers found for a variety of $m$ within the range [1.05, 9.0]; the clusters found with $m = 2$ are connected with the thin dotted line. It should be noted that, following our experiments, DE converges more successfully than the standard FCM within the considered range of $m$. It can be seen from Fig. 3(a) and (b) that in the case when $m_1 = m_2$, i.e. for type-1 FCM the cluster centers are located within the bounds of the cluster squares obtained by type-2 FCM. At the same time, the cluster squares allows capturing more uncertainty in data than one-point cluster centers.

Even for narrower ranges of $m$, say $m = [1.4, 2.6]$ DE as a global search algorithm is expected to be more advantageous that standard FCM for the case of large number of highly-dimensional data vectors. To show this, we run the following experiment. We generate 1,000 data vectors of dimension 10 on the basis of 7 original vectors C1, C2, C3, C4, C5, C6, C7:

| $j$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| C1= | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 4.0 | 3.0 | 2.0 | 1.0 | 2.0 |
| C2= | 2.0 | 7.0 | 8.0 | 3.0 | 7.0 | 8.0 | 4.0 | 7.0 | 8.0 | 5.0 |
| C3= | 1.0 | 1.0 | 1.5 | 1.0 | 1.0 | 1.5 | 1.0 | 1.0 | 1.0 | 1.0 |
| C4= | 4.0 | 6.0 | 7.0 | 5.0 | 8.0 | 3.0 | 5.0 | 4.0 | 9.0 | 9.0 |
| C5= | 2.0 | 7.5 | 8.0 | 3.0 | 7.5 | 8.0 | 4.0 | 7.0 | 8.0 | 4.5 |
| C6= | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 2.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| C7= | 1.0 | 6.0 | 3.0 | 3.0 | 5.0 | 5.0 | 2.0 | 6.0 | 6.0 | 8.0 |

as follows:

$$p_{ik} = C_{((i-1)\%7+1),k} + 1.5 \cdot (Rand() - 0.5); \quad i = 1, 2, \ldots, 1000; \quad k = 1, 2, \ldots, 10,$$

where Rand () is a random number drawn from the uniform distribution [0, 1] and % is the operation of determining a remainder of integer division.

Consecutive 5 runs of the FCM algorithm applied to the generated 1,000 for finding 7 cluster centers with $m = 2.5$ have ended up with the following values of the objective function (5) after 1,000 iterations (at which point the FCM has converged and no more iterations are required):

| Experiment | Objective function ($J_{m=2.5}$) |
|---|---|
| 1 | 88.148 |
| 2 | 96.902 |
| 3 | 84.988 |
| 4 | 94.693 |
| 5 | 118.788 |
| 6 | 84.988 |

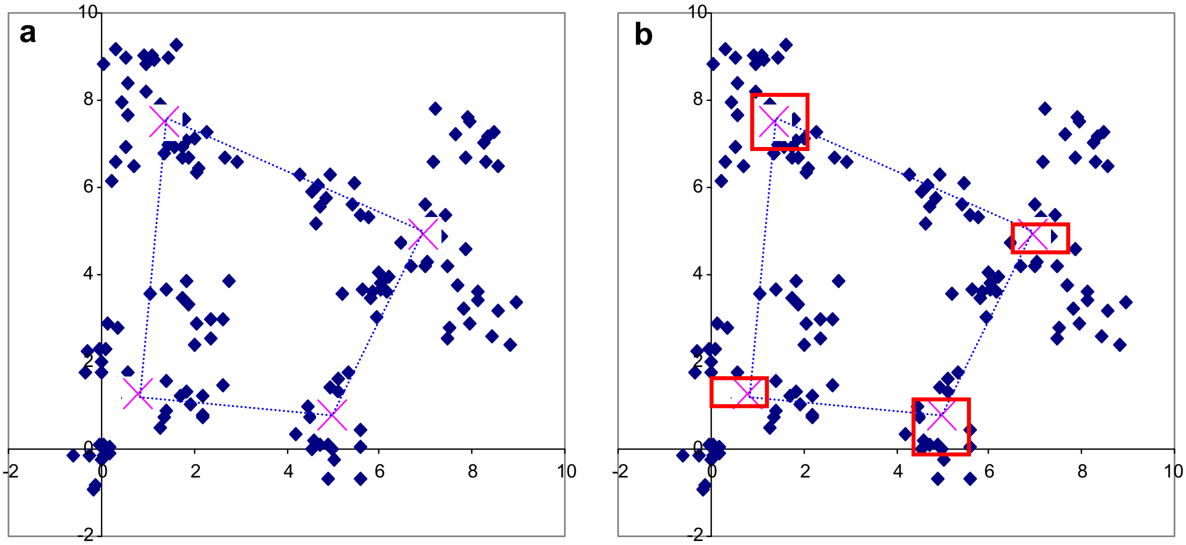


**Fig. 3.** Clustering results produced by the FCM (a) and the DE-based type-2 fuzzy clustering (b).

The application of the global search DE algorithm ($ps$ = 200) allowed us to minimize the objective function (5) down to $J_{m=2.5}$ = 79.076. The clusters found by the DE based clustering are shown below:

| $J$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| $\mathbf{v}_1$ | 1 | 1.95 | 2.98 | 4.01 | 4.97 | 3.98 | 3 | 2.01 | 1 | 1.98 |
| $\mathbf{v}_2$ | 1.98 | 7.05 | 7.98 | 2.99 | 6.99 | 7.97 | 4 | 6.96 | 7.94 | 5.01 |
| $\mathbf{v}_3$ | 1.99 | 7.46 | 8.02 | 2.99 | 7.48 | 7.96 | 4 | 6.96 | 8.06 | 4.5 |
| $\mathbf{v}_4$ | 3.99 | 5.97 | 6.99 | 4.95 | 8.01 | 3 | 4.95 | 3.99 | 8.98 | 8.98 |
| $\mathbf{v}_5$ | 0.99 | 5.99 | 3.01 | 2.99 | 4.99 | 4.99 | 2.01 | 5.99 | 6 | 8 |
| $\mathbf{v}_6$ | 0.99 | 0.99 | 1.49 | 0.99 | 0.99 | 1.5 | 1 | 0.99 | 1.01 | 1 |
| $\mathbf{v}_7$ | 1 | 0.99 | 1.01 | 0.99 | 0.99 | 1.98 | 0.99 | 0.98 | 0.99 | 0.99 |

Note that the actual minimum value of the objective function for this example (found by the optimization with initial cluster centers set to the original 7 data vectors instead of random numbers) is approximately 77.7.

The parameters to be optimized by the DEO when minimizing the objective function are the centers of the clusters, $\mathbf{v}_i^{(1)},\ \mathbf{v}_i^{(2)},\ i = 1, 2, \ldots, c$. The DE-based clustering algorithm can be formally described as follows:

1. Set the number of clusters $c = 2$ (start with the minimal number of clusters, that is $c = 2$);
2. Compute cluster centers for $m_1:\ \mathbf{v}_i^{(1)}\ (i = 1, 2, \ldots, c)$
   - 2.1. Create a population of random solutions (population size is $ps$), i.e., cluster centers $\mathbf{v}_i^{(1)}:\ P = \{\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_{ps}\}$ (Each $\mathbf{S}_j$ ($j$ = 1,2,…, $ps$) represent a combination of cluster centers for given data).
   - 2.2. Randomly select 4 different trial vectors from $P$: $\mathbf{S}_{r1}$, $\mathbf{S}_{r2}$, $\mathbf{S}_{r3}$, $\mathbf{S}_{r4}$, $1 \leqslant r_1 \neq r_2 \neq r_3 \neq r_4 \leqslant ps$.
   - 2.3. Compute the new trial solution as $\mathbf{S}_n = (\mathbf{S}_{r1} - \mathbf{S}_{r2})f + \mathbf{S}_{r3}$.
   - 2.4. Set the cost function to $J_{m1}$. Compute the cost value of $\mathbf{S}_n$. If Cost ($\mathbf{S}_n$) < Cost ($\mathbf{S}_{r4}$), replace $\mathbf{S}_{r4}$ by $\mathbf{S}_n$.
   - 2.5. If the maximal number of generations has not been reached, go to Step 2.2, else go to step 3.
   - 2.6. Find the solution with the minimum cost value, $\mathbf{S} = \mathbf{S}_{best}$, where Cost ($\mathbf{S}_{best}$) $\leqslant$ Cost ($\mathbf{S}_r$), $r$ = 1,2,…,$ps$. Retrieve optimal cluster centers $\mathbf{v}_i^{(1)}$ from $\mathbf{S}_{best}$.
3. Compute cluster centers for $m_2:\ \mathbf{v}_i^{(2)}\ (i = 1, 2, \ldots, c)$
   - 3.1. Create the population of random solutions (cluster centers $\mathbf{v}_i^{(2)}$) : $P = \{\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_{ps}\}$.
   - 3.2. Randomly select 4 different trial vectors from $P$: $\mathbf{S}_{r1}$, $\mathbf{S}_{r2}$, $\mathbf{S}_{r3}$, $\mathbf{S}_{r4}$, $1 \leqslant r_1 \neq r_2 \neq r_3 \neq r_4 \leqslant ps$
   - 3.3. Compute the new trial solution as $\mathbf{S}_n = (\mathbf{S}_{r1} - \mathbf{S}_{r2})f + \mathbf{S}_{r3}$.
   - 3.4. Set the cost function to $J_{m2}$. Compute the cost value of $\mathbf{S}_n$. If Cost ($\mathbf{S}_n$) < Cost ($\mathbf{S}_{r4}$), replace $\mathbf{S}_{r4}$ by $\mathbf{S}_n$.
   - 3.5. If the number of generations has not reached go to Step 3.2, else go to step 4
   - 3.6. Find the solution with minimum cost value $\mathbf{S} = \mathbf{S}_{\text{best}}$, where Cost ($\mathbf{S}_{best}$) $\leqslant$ Cost ($\mathbf{S}_r$), $r = 1, 2, \ldots, ps$. Retrieve optimal cluster centers $\mathbf{v}_i^{(2)}$ from $\mathbf{S}_{best}$.

4. Check if the validity criterion $V = \min(V_{m1}, V_{m2})$ attains higher value than a predefined threshold (e.g., 0.6). If yes, then go to step 5, otherwise increase $c$ by one ($c = c + 1$) and repeat from step 2.
5. On the basis of optimal cluster centers $\mathbf{v}_i^{(1)}$ and $\mathbf{v}_i^{(2)}$ form the interval cluster centers: $\tilde{\mathbf{v}}_i = \left[\min\left(\mathbf{v}_i^{(1)}, \mathbf{v}_{Ind_i}^{(2)}\right), \max\left(\mathbf{v}_i^{(1)}, \mathbf{v}_{Ind_i}^{(2)}\right)\right]$, where $Ind_i = \arg\min_j \left\|\mathbf{v}_i^{(1)}, \mathbf{v}_j^{(2)}\right\|$.
6. Stop

The clusters produced by the DEO-based clustering technique are used to supply fuzzy rules with initial membership functions to the Type-2 Fuzzy Inference Neural Network. Afterwards we realize further refinement of the parameters of the neural system to arrive at a sound trade-off between transparency and accuracy of the resulting model.

## 5. Type-2 fuzzy inference neural network

### 5.1. The structure of type-2 fuzzy inference neural network

The structure of the proposed Type-2 Fuzzy Inference Neural Network (T2FINN) is shown in Fig. 4. Let us elaborate on the functionality of the layers in more detail. Layer 1 consists of fuzzifiers that map inputs to type-2 fuzzy terms used in the rules. Layer 2 comprises nodes representing these rules. Each rule node performs the Min operation on the outputs (interval valued membership degrees) of the incoming links from the previous layer. Layer 3 consists of output terms membership functions of type-1. Layer 4 computes the fuzzy output signal for the output variables. Layer 5 realizes the defuzzification using the Center-of-Gravity (COG) defuzzification.

### 5.2. Fuzzification and inference procedure

The proposed T2FINN uses an arbitrary number of type-2 fuzzy input variables and an arbitrary number of type-1 fuzzy output variables. The input variable's type-2 fuzzy terms are described as:

$$\widetilde{A} = \{x/\mu_x\}, \quad x \in U \subset \Re,$$
$$\mu_x = \{\alpha/1\}, \quad \alpha \in M_x \subset [0,1],$$

where:

$$M_x = \begin{cases} [\alpha_{x1}, \alpha_{x2}], & \text{if } \left(\frac{\alpha_{x1}+\alpha_{x2}}{2} < \frac{\alpha_{x3}+\alpha_{x4}}{2}\right), \\ [\alpha_{x3} + \alpha_{x4}], & \text{elsewise} \end{cases}$$

and

$$\alpha_{x1} = \max\left(\min\left(1, \frac{RL - x}{RL - ML}\right), 0\right),$$
$$\alpha_{x2} = \max\left(\min\left(1, \frac{RR - x}{RR - MR}\right), 0\right),$$
$$\alpha_{x3} = \max\left(\min\left(1, \frac{x - LL}{ML - LL}\right), 0\right),$$
$$\alpha_{x4} = \max\left(\min\left(1, \frac{x - LR}{MR - LR}\right), 0\right).$$

Here $LL$, $LR$, $ML$, $MR$, $RL$, $RR$ ($LL \leqslant LR \leqslant ML \leqslant MR \leqslant RL \leqslant RR$) are parameters defining the "shape" of the type-2 fuzzy membership functions. An example of type-2 fuzzy value defined using this type-2 fuzzy number representation (say, [0.25, 0.75], [1.25, 1.75], [2.25, 3.00]) is shown in Fig. 4. As it can be seen, this type-2 fuzzy number can be composed of three intervals, say $[LL, LR]$ (a left interval, identified by letter L in the figure), $[ML, MR]$ (a medium interval, indicated by letter M), and $[RL, RR]$ (a right interval, denoted by letter R): $[LL, LR]$, $[ML, MR]$, $[RL, RR]$. The input term membership functions can be considered as interval-valued membership functions (interval membership values for two values of $x$ are shown: $x = 1$ and $x = 2.5$).

The output variable's type-1 fuzzy terms are type-1 fuzzy trapezoidal numbers taking on the form:

$$B = [L, ML, MR, R] = [[L, L], [ML, MR], [R, R]].$$

We used the Zadeh's implication to compute the output membership functions. As a result, for every output variable, we obtain two piecewise linear membership functions:

$$\tilde{y}_i = \{y/[\mu_{Li}(y), \mu_{Ri}(y)]\}.$$

Type reduction is performed on the basis of the center of gravity (COG) defuzzification procedure, that is

$$COG(\tilde{y}_i) = \{y/[COG(\mu_{Li}(y)), COG(\mu_{Ri}(y))]\} = \{y/[y_{Li}, y_{Ri}]\}.$$

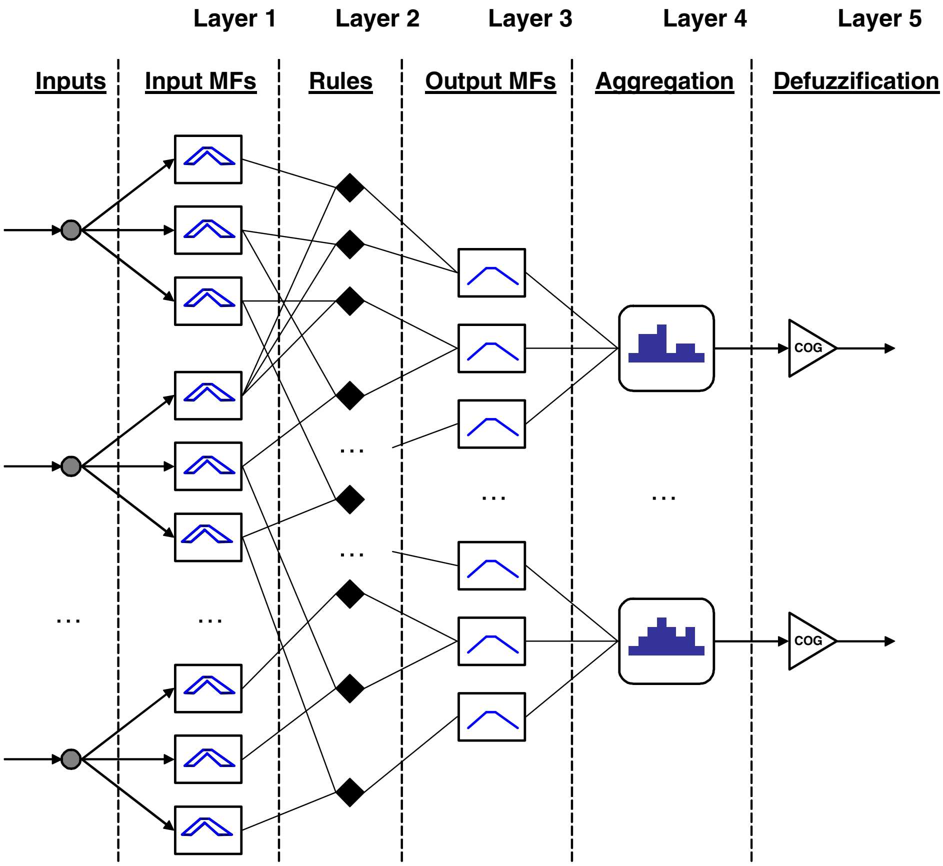


**Fig. 4.** Structure of type-2 fuzzy inference neural network.

The final defuzzification is done through averaging the two values:

$$y_i = \frac{y_{Li} + y_{Ri}}{2}.$$

In what follows, we discuss the implementation of of the COG defuzzification used for type-reducing of aggregated output type-2 fuzzy values (with piecewise linear membership functions).

*5.3. The COG defuzzification for fuzzy sets with piecewise linear membership functions*

Fuzzy sets with piecewise membership functions can be produced, for example, as a result of fuzzy inference when fuzzy terms with trapezoidal membership functions are used in both in the antecedent and consequent parts of fuzzy "If-Then" rules

$$A = \{x_k/\alpha_k\}, \quad k = 1, 2, \ldots, K,$$
$$\alpha_k = \mu_A(x_k).$$

By definition:

$$A_{defuzz} = \frac{\int_{x_1}^{x_K} x\mu_A(x)dx}{\int_{x_1}^{x_K} \mu_A(x)dx}.$$

Any (linear) part of $\mu_A(x)$ in an interval $[x_k, x_{k+1}]$ can be represented as:

$$\mu_A(x) = \frac{\alpha_{k+1} - \alpha_k}{x_{k+1} - x_k}x + \left(\alpha_k - \frac{\alpha_{k+1} - \alpha_k}{x_{k+1} - x_k}x_k\right).$$

Then, the above integrals are calculated as:

$$\int_{x_k}^{x_{k+1}} \mu_A(x)dx = \frac{(\alpha_k + \alpha_{k+1})(x_{k+1} - x_k)}{2},$$
$$\int_{x_k}^{x_{k+1}} x\mu_A(x)dx = \frac{(2\alpha_k x_k + 2\alpha_{k+1}x_{k+1} + \alpha_k x_{k+1} + \alpha_{k+1}x_k)(x_{k+1} - x_k)}{6}.$$

For a single interval $[x_k, x_{k+1}] : A_{defuzz} = \frac{(2\alpha_k x_k + 2\alpha_{k+1}x_{k+1} + \alpha_k x_{k+1} + \alpha_{k+1}x_k)}{3(\alpha_{k+1} + \alpha_k)}$.

For the entire domain $[x_1, x_K]$ we obtain:

$$A_{defuzz} = \frac{1}{3}\frac{\sum_{k=1}^{K-1}(2\alpha_k x_k + 2\alpha_{k+1}x_{k+1} + \alpha_k x_{k+1} + \alpha_{k+1}x_k)(x_{k+1} - x_k)}{\sum_{k=1}^{K-1}(\alpha_k + \alpha_{k+1})(x_{k+1} - x_k)}.$$

### *5.4. Training of T2FINN and the optimization of rule base*

Once we have obtained the initial rule base by type-2 fuzzy clustering, we need to optimize the parameters of the rule base. To optimize these parameters a certain error function is considered as shown below

$$E = \frac{1}{n \cdot s_y}\sum_{p=1}^{n}\sum_{i=1}^{s_y}(y_{pi}^* - y_{pi})^2. \tag{9}$$

Here $y_{pi}^*$ is the desired value (target) for output $i$ when we apply input value vector $\mathbf{x}_p$, $y_{pi}$ is the corresponding output of the model output, $n$ is the number of training patterns, and $s_y$ is the number of outputs in the model (typically being equal to 1).

To arrive at the minimal value of error of the T2FINN, see (9), we use differential evolution (DE). The parameters *LL*, *LR*, *ML*, *MR*, *RL*, *RR* for all input terms and the parameters *L*, *ML*, *MR*, *R* for all outputs terms are adjusted.

The DEO-based parameter optimization for the T2FINN is summarized as follows:

Step 0. Initialize DE
- Step 0.1. Construct template parameter vector $S$ of dimension ($N_{par}$) necessary to accommodate all parameters to be optimized. $X$ consists of the complete set of parameters (*LL*, *LR*, *ML*, *MR*, *RL*, or *RR*) for all input terms and the complete set of parameters (*L*, *ML*, *MR*, or *R*) for all output terms.
- Step 0.2. Set the essential parameters of the DE: $f$ (mutation rate), *cr* (crossover rate), and *ps* (size of population).
- Step 0.3. Define the cost (objective) function expressed as the Mean Square Error (MSE) between the output of the fuzzy model with the current parameters and desired output (coming from the training data): $\text{Cost} = E = \frac{1}{n \cdot s_y}\sum_{p=1}^{n}\sum_{i=1}^{s_y}\left(y_{pi}^* - y_{pi}\right)^2$.

Step 1. Randomly generate *ps* parameter vectors (from respective parameter spaces, e.g., in the range $[-1,1]$) and form a population $P = \{\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_{ps}\}$.

Step 2. While the termination condition (say, the number of predefined generations or the required error level) has not been reached, generate new parameter sets:
- Step 2.1. Choose a next vector $\mathbf{S}_i$ ($i = 1, \ldots, ps$).
- Step 2.2. Choose randomly three different vectors from $P$: $\mathbf{S}_{r1}$, $\mathbf{S}_{r2}$, $\mathbf{S}_{r3}$ ($1 \leqslant r_1 \neq r_2 \neq r_3 \leqslant ps$) each of which is different from the current $\mathbf{S}_i$.
- Step 2.3. Generate a trial vector $\mathbf{S}_t = (\mathbf{S}_{r1} - \mathbf{S}_{r2})f + \mathbf{S}_{r3}$.
- Step 2.4. Generate a new vector from the trial vector $\mathbf{S}_t$. Individual vector parameters of $\mathbf{S}_t$ are inherited with probability *cr* and included into the new vector $\mathbf{S}_{new}$. If the cost function from $\mathbf{S}_{new}$ is better (that is lower) than the cost function from $\mathbf{S}_i$, current $\mathbf{S}_i$ is replaced in population $P$ by $\mathbf{S}_{new}$.

Next i.

Step 3. Select the parameter vector $\mathbf{S}_{best}$ (best fuzzy model parameter set) with the lowest cost (= MSE) function from population $P$. Extract from $\mathbf{S}_{best}$ all parameters for defining adjusted type-2 membership functions in the rules, i.e. the parameters *LL*, *LR*, *ML*, *MR*, *RL*, *RR* for all input terms and the parameters *L*, *ML*, *MR*, *R* for all outputs terms.

Step 4. Terminate the algorithm.

## 6. Computer simulations

### *6.1. Estimation of non-linear model*

As a simple, yet highly illustrative example, let us consider a certain numeric mapping as shown in Fig. 5. Assume that no explicit model is available – just only a few rules are known that are provided by an expert. Other numerical input–output data that describe the relationship are uncertain or contain noise. We consider that any numeric data is affected by noise and comes with a level of error of up to 10%. In this case, an available datum $v$ can be adequately expressed as a certain interval being reflective of the associated error, that is [0.95v, 1.05v].

Let us assume that the linguistic rules coming from the expert are of the form:

**IF** X **IS** “VERY LOW” **THEN** Y **IS** “LOW”
**IF** X **IS** “LOW” **THEN** Y **IS** “HIGH”
**IF** X **IS** “HIGH” **THEN** Y **IS** “VERY LOW”
**IF** X **IS** “VERY HIGH” **THEN** Y **IS** “VERY HIGH”

Here X and Y represent input and output variables, respectively. X takes values from type-2 fuzzy termset: A = {“VERY LOW”(X), “LOW”(X), “HIGH”(X), “VERY HIGH”(X)} and Y takes value from type-1 fuzzy termset B = {“VERY LOW”(Y), “LOW”(Y), “HIGH”(Y), “VERY HIGH”(Y)}.

The initial input type-2 fuzzy terms in the form $\widetilde{A}_i = [[LL_i, LR_i], [ML_i, MR_i], [RL_i, RR_i]]$ and output type-1 fuzzy terms in form $B_i = [L_i, ML_i, MR_i, R_i]$ are generated manually by help of the expert. The initial terms can be drawn by precisiation of the available linguistic information using the considered type-1 (ordinary trapezoid) and type-2 (type-2 trapezoid: see Fig. 5) fuzzy number representation. The construction of type-2 fuzzy terms is done on the basis of some hints about the width of FOU of type-2 membership function generated by the expert.

The initial defuzzified input/output relationship of the system produced on the basis of expert given rules and initial termsets is shown in Fig. 6.

The final membership functions of A and B obtained after the network has been trained with the use of the DE optimization come in the form:

For X:
VERY LOW = [ [ 0.00, 0.30 ], [ 2.09, 2.12 ], [ 3.16, 4.00 ] ],
LOW = [ [ 3.57, 3.82 ], [ 4.35, 4.54 ], [ 4.84, 5.00 ] ],
HIGH = [ [ 1.02, 2.60 ], [ 2.60, 2.69 ], [ 2.70, 2.95 ] ],
VERY HIGH = [ [ 1.42, 1.77 ], [ 2.03, 2.33 ], [ 2.72, 3.12 ] ].
For Y:
VERY LOW = [ [ 2.03, 2.03 ], [ 4.65, 5.30 ], [ 7.81, 7.81 ] ],
LOW = [ [ 0.00, 0.00 ], [ 0.01, 0.56 ], [ 1.84, 1.84 ] ],
HIGH = [ [ 3.57, 3.57 ], [ 4.07, 8.08 ], [ 10.00, 10.00 ] ],
VERY HIGH = [ [ 1.85, 1.85 ], [ 3.58, 5.06 ], [ 7.80, 7.80 ] ].

Fig. 7 includes the comparison of the actual output of the system being identified and the output generated by the proposed T2FINN. About 10 software runs were performed to check the learning effect. Membership functions obtained in all experiments were very similar, which is an indication of robustness of the system. The MSE obtained by the DE learning in best software run was $4.71 \cdot 10^{-4}$ (for comparison, the MSE obtained by experiments with similar system (the same number of rules and dimensions of term-sets) based on type-1 logic was $1.03 \cdot 10^{-3}$).

### 6.2. Nonlinear system identification

This benchmark problem for nonlinear system identification was taken from [8]. The dynamic system is governed by the following equation:

$$x(k) = g(x(k-1), x(k-2)) + u(k) \tag{10}$$

where

$$g(x(k-1), x(k-2)) = \frac{x(k-1)x(k-2)(x(k-1)-0.5)}{1 + x^2(k-1) + x^2(k-2)} \tag{11}$$

The goal is to construct the T2FINN for the non-linear system (10) and (11). 200 input–output data pairs were used for training and 200 others for testing.

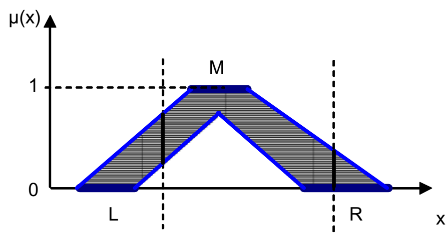


**Fig. 5.** A type-2 fuzzy term value.

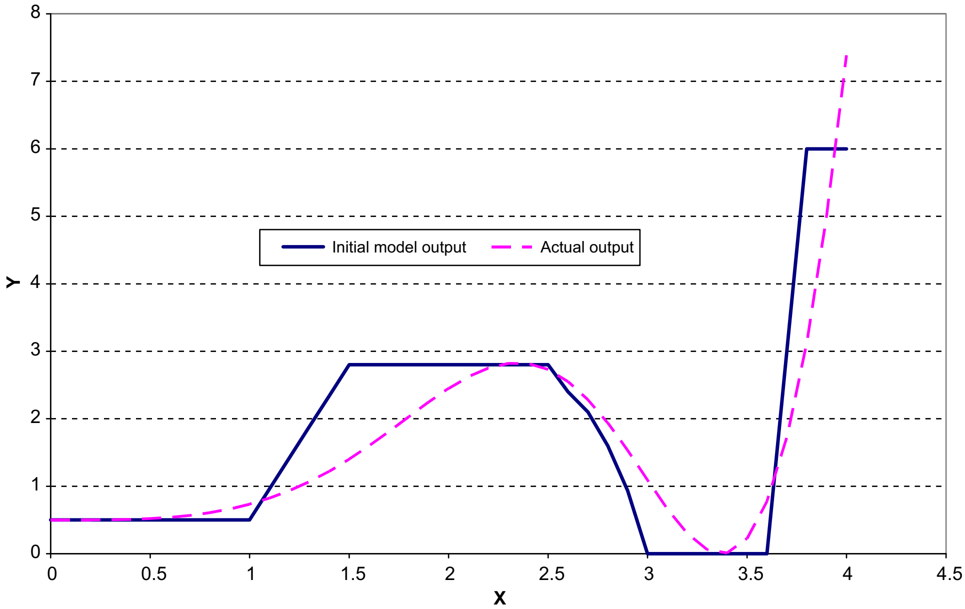


**Fig. 6.** Input/output relationship produced by T2FINN with the use of initial rules (termset) vis-à-vis the actual curve.

After applying the DE-based clustering, we arrive at the following 5 rules.

**IF** $x(k-2)$ **IS** A1 **AND** $x(k-1)$ **IS** B1 **THEN** $x(k)$ **IS** C1
**IF** $x(k-2)$ **IS** A2 **AND** $x(k-1)$ **IS** B2 **THEN** $x(k)$ **IS** C2
**IF** $x(k-2)$ **IS** A3 **AND** $x(k-1)$ **IS** B3 **THEN** $x(k)$ **IS** C3
**IF** $x(k-2)$ **IS** A4 **AND** $x(k-1)$ **IS** B4 **THEN** $x(k)$ **IS** C4
**IF** $x(k-2)$ **IS** A5 **AND** $x(k-1)$ **IS** B5 **THEN** $x(k)$ **IS** C5

The progression of the DE optimization is quantified in terms of the values of the MSE reported in successive iterations, see Fig. 8. It becomes noticeable that the reduction of the error is quite fast and the optimization proceeds smoothly without any significant oscillations.

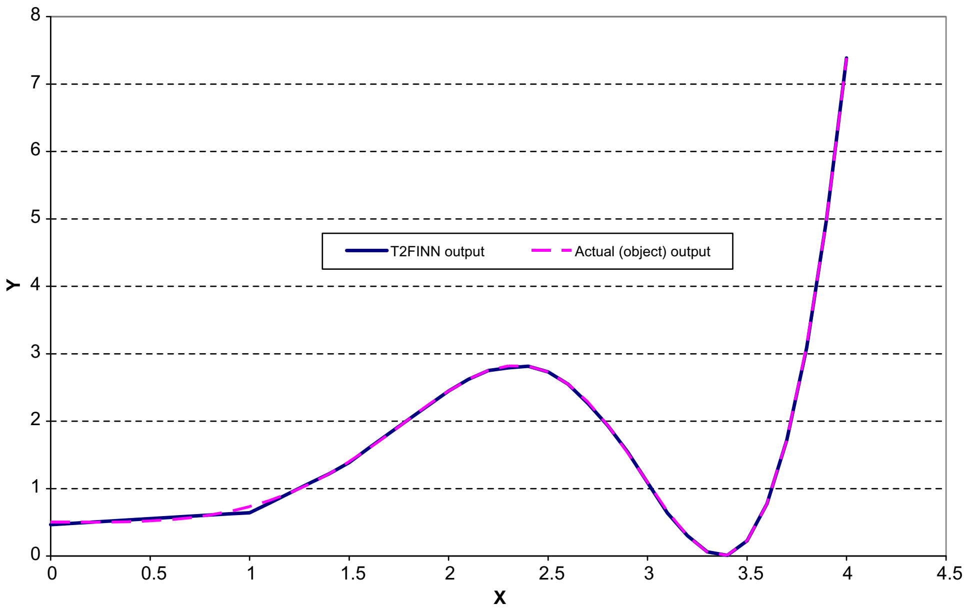


**Fig. 7.** Input/output relationship produced by T2FINN after the DE-based training.

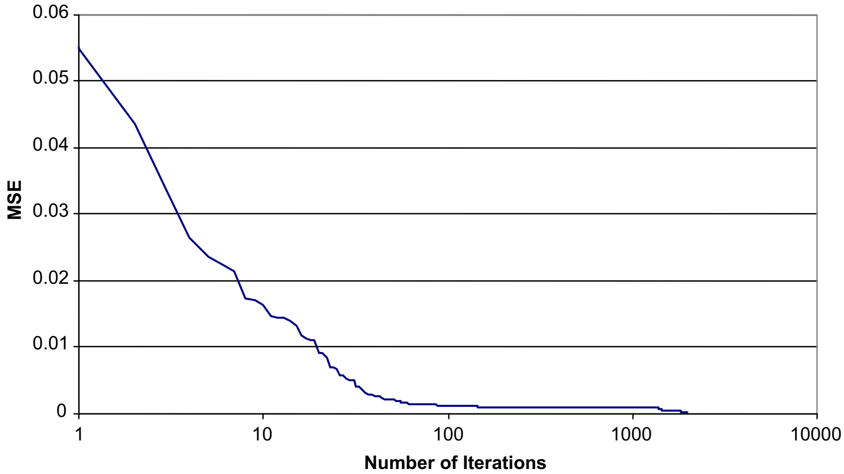


**Fig. 8.** Minimized performance index in successive iterations of the DE optimization.

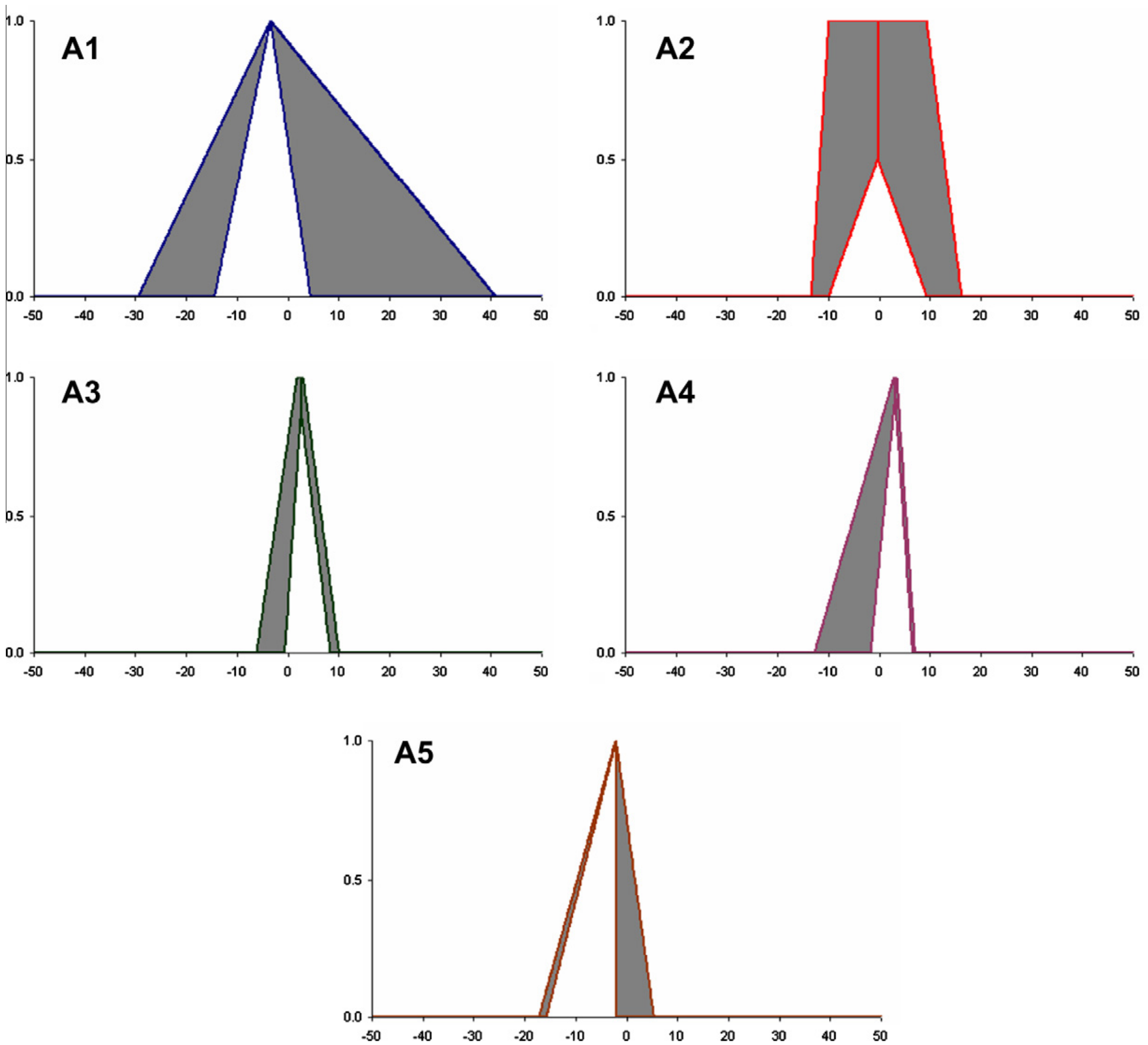


**Fig. 9.** Final FOUs.

**Table 1**
MSE values produced by different versions of the network.

| | MSE (train data) | MSE (test data) |
|---|---|---|
| Fuzzy model [8] (type-1) | $1.90 \cdot 10^{-4}$ | $3.80 \cdot 10^{-4}$ |
| Type-1 FINN | $5.40 \cdot 10^{-4}$ | $9.50 \cdot 10^{-4}$ |
| Type-2 FINN | $2.11 \cdot 10^{-4}$ | $2.84 \cdot 10^{-4}$ |

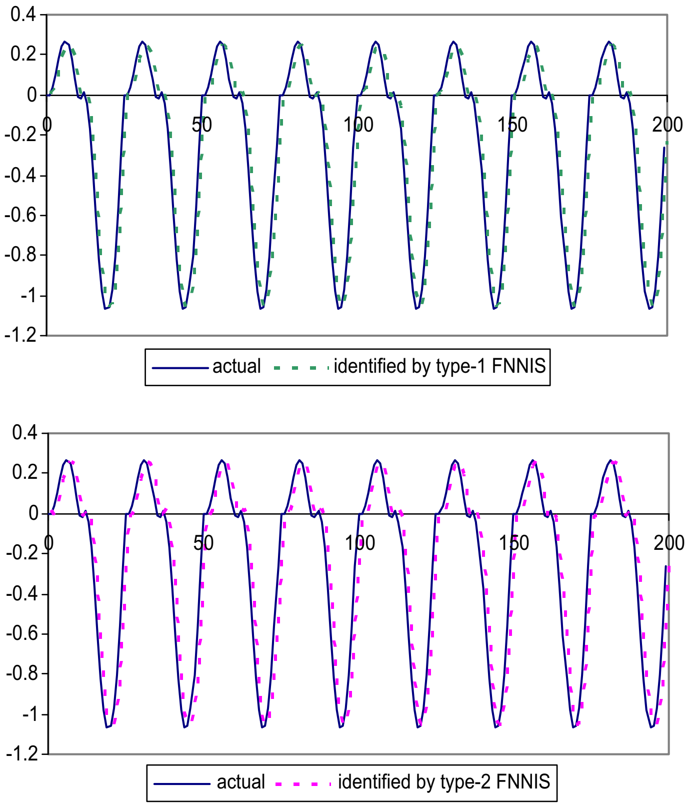


**Fig. 10.** The actual system response and the output of the model.

The type-2 fuzzy terms obtained after the training are the following:

Type-2 fuzzy terms for $x(k-2)$:
A1 = [ [ −29.46, −14.39 ], [ −3.38, −3.38 ], [ 4.50, 40.91 ] ]
A2 = [ [ −13.40, −9.95 ], [ −9.89, 9.39 ], [ 9.39, 16.29 ] ]
A3 = [ [ −6.12, −0.58 ], [ 1.84, 3.04 ], [ 8.37, 10.12 ] ]
A4 = [ [ −12.70, −1.69 ], [ 2.93, 3.50 ], [ 6.60, 7.09 ] ]
A5 = [ [ −17.17, −15.70 ], [−2.12, −2.03 ], [ −2.03, 5.40 ] ]

Type-2 fuzzy terms for $x(k-1)$
B1 = [ [ −20.98, −20.72 ], [ 3.18, 4.91 ], [ 35.59, 38.01 ] ]
B2 = [ [ −16.90, −1.38 ], [ −0.83, 2.19 ], [ 2.43, 6.73 ] ]
B3 = [ [ −17.95, −8.29 ], [ −2.78, −1.58 ], [ 0.10, 1.59 ] ]
B4 = [ [ −2.03, −1.37 ], [ −0.95, 1.40 ], [ 1.41, 1.52 ] ]
B5 = [ [ −8.96, −3.84 ], [ −3.57, −0.30 ], [ 10.84, 15.49 ] ]

Type-2 fuzzy terms for x (k):
C1 = [ [ −6.77, −6.77 ], [ −6.74, −1.86 ], [ 15.46, 15.46 ] ]
C2 = [ [ −5.92, −5.92 ], [ −2.01, 9.87 ], [ 13.53, 13.53 ] ]
C3 = [ [ −10.28, −10.28 ], [ −0.81, 2.08 ], [ 22.32, 22.32 ] ]
C4 = [ [ −15.85, −15.85 ], [ −4.91, 2.10 ], [ 9.80, 9.80 ] ]
C5 = [ [ −19.73, −19.73 ], [ −14.11, −1.69 ], [ 8.39, 8.39 ] ]

For further visualization, the obtained FOUs for the type-2 fuzzy terms A1, A2, A3, A4, and A5 after training are illustrated in Fig. 9.

The values of the MSE obtained on the training data and on the testing data for the best of 5 software experiments were equal to $2.11 \cdot 10^{-4}$ and $2.84 \cdot 10^{-4}$, respectively. For comparison, as reported in [8], the MSE for the same problem was $1.90 \cdot 10^{-4}$ (training data) and $3.80 \cdot 10^{-4}$ (testing data). For the type-1 fuzzy based system, the best result obtained for the same problem was $9.50 \cdot 10^{-4}$ on the testing data. All the results are concisely summarized in Table 1.

The comparison of the actual response of the system and the output of the model is included in Fig. 10 (the results are shown for testing data). Good performance on testing data validates that the method is not only accurate but also robust.

## 7. Applications

In this section, we present several applications of the designed network by showing its performance in the problem of petrol production forecasting and customer credit evaluation. We stress here the interpretability capabilities of type-2 fuzzy sets used in the framework of system modeling.

### 7.1. Petrol production forecasting

The problem is to forecast petrol (A92) production for optimal scheduling of oil refinery plant. In the fuzzy forecasting model we used 3 inputs and 1 output where the predicted variable is associated with the historical data $x(k)$, $x(k-1)$, and $x(k-2)$:

$$x(k+1) = F(x(k-2), x(k-1), x(k)).$$

For this example, we considered the actual daily data coming from existing petrol production unit and covering one month period. The fuzzy prediction model was built on the basis of type-2 fuzzy "If-Then" rules with 3 type-2 fuzzy variables in antecedent and 1 in consequent part. Approximately 80% of the data (selected randomly) were used for training the network and the remaining data were used for testing purposes. By applying the fuzzy clustering for the available training data, it was concluded, on the basis of the selected validation criterion (6), that the optimal number of clusters is 4. Therefore 4 rules with 4 membership functions for each input and output variables' termsets were used as the model's initial knowledge base.

The clustering algorithm produced the cluster centers shown in Table 2.

The rules come in the form:

**IF** $x(k-2)$ **IS** A1 **AND** $x(k-1)$ **IS** B1 **AND** $x(k)$ **IS** C1 **THEN** $x(k+1)$ **IS** D1;
**IF** $x(k-2)$ **IS** A2 **AND** $x(k-1)$ **IS** B2 **AND** $x(k)$ **IS** C2 **THEN** $x(k+1)$ **IS** D2;
**IF** $x(k-2)$ **IS** A3 **AND** $x(k-1)$ **IS** B3 **AND** $x(k)$ **IS** C3 **THEN** $x(k+1)$ **IS** D3;
**IF** $x(k-2)$ **IS** A4 **AND** $x(k-1)$ **IS** B4 **AND** $x(k)$ **IS** C4 **THEN** $x(k+1)$ **IS** D4.

The initial type-2 fuzzy terms A1, A2, A3, A4 were formed from the component $x(k-2)$ of the cluster vectors 1, 2, 3, and 4, respectively:

A1 = [ [ 12.30, 12.30 ], [ 12.30, 28.93 ], [ 28.93, 28.93 ] ];
A2 = [ [ 30.72, 30.72 ], [ 30.72, 38.81 ], [ 38.81, 38.81 ] ];
A3 = [ [ 44.03, 44.03 ], [ 44.03, 52.25 ], [ 52.25, 52.25 ] ];
A4 = [ [ 47.48, 47.48 ], [ 37.49, 47.48 ], [ 47.48, 47.48 ] ].

Similarly, we formed the terms B1, B2, B3, B4, C1, C2, C3, C4, D1, D2, D3, and D4.

The MSE values obtained for the training data and testing data were equal to $1.2 \cdot 10^{-3}$ and 1.57, respectively.

The type-2 fuzzy terms obtained after training come as follows:

A1 = [ [ 0.00, 7.97 ], [ 12.300, 25.53 ], [ 52.00, 59.00 ] ]
A2 = [ [ 4.14, 16.24 ], [ 20.86, 24.62 ], [ 47.11, 57.19 ] ]
A3 = [ [ 2.42, 5.63 ], [ 10.47, 31.87 ], [ 54.47, 57.60 ] ]
A4 = [ [ 9.23, 15.75 ], [ 18.79, 62.92 ], [ 66.56, 69.03 ] ]

**Table 2**
Cluster centers (prototypes) obtained for petrol production data.

| Variables\cluster centers | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| $x(k-2)$ | [12.30, 28.93] | [30.72, 38.81] | [44.03, 52.25] | [37.49, 47.48] |
| $x(k-1)$ | [38.02, 38.59] | [40.06, 41.36] | [35.71, 36.24] | [37.63, 38.21] |
| $x(k)$ | [36.88, 37.87] | [38.93, 40.11] | [40.96, 41.18] | [20.45, 23.24] |
| $x(k+1)$ | [40.48, 43.61] | [15.06, 19.97] | [30.67, 33.40] | [46.77, 49.82] |

B1 = [ [ 0.00, 1.20 ], [ 2.16, 24.64 ], [ 44.00, 52.00 ] ]
B2 = [ [ 42.44, 49.21 ], [ 57.37, 70.72 ], [ 75.77, 79.99 ] ]
B3 = [ [ 15.58, 18.32 ], [ 24.68, 28.75 ], [ 29.07, 34.99 ] ]
B4 = [ [ 31.42, 32.00 ], [ 35.07, 39.84 ], [ 40.03, 48.99 ] ]

C1 = [ [ 0.00, 23.99 ], [ 32.52, 40.87 ], [ 75.45, 77.99 ] ]
C2 = [ [ 7.99, 36.45 ], [ 57.75, 60.80 ], [ 64.48, 79.99 ] ]
C3 = [ [ 17.95, 33.72 ], [ 35.13, 41.98 ], [ 50.74, 79.99 ] ]
C4 = [ [ 23.08, 34.00 ], [ 40.36, 49.68 ], [ 58.17, 79.99 ] ]

D1 = [ [ 12.13, 12.13 ], [ 40.28, 75.31 ], [ 79.99, 79.99 ] ]
D2 = [ [ 12.10, 12.10 ], [ 34.43, 37.44 ], [ 41.99, 41.99 ] ]
D3 = [ [ 9.29, 9.29 ], [ 18.54, 50.42 ], [ 56.68, 56.68 ] ]
D4 = [ [ 0.00, 0.00 ], [ 0.00, 0.00 ], [ 31.01, 31.01 ] ]

The used type-2 fuzzy clustering method provides better location of the cluster centers, and subsequently results in a better fuzzy rule model. This in turn allows capturing more uncertainty and deliver higher robustness against the imprecision of the data.

### *7.2. Customer credit evaluation*

The credit risk evaluation is a major problem present in the area in banking and finance. It is extremely difficult do determine whether or not a granted loan will return in due time or will be returned at all. Many banks have developed methods to evaluate credibility of their clients before granting loans but they never have been fully satisfied with the end result. There are many studies on client evaluation using intelligent techniques. These techniques mostly involve rule based construction in which fuzzy reasoning is the leading method. The rule base is related with uncertainty implied by a variety of survey procedures being used to mine knowledge from experts. So far, there has been no research done by using type-2 fuzzy logic approach to loan assessment. Here, we look at this modeling alternative. The proposed system has been applied to a sample of real data received from a local bank in Azerbaijan.

In the considered case, the main factors taken into consideration in credit analysis are: credit history, age, net income (salary), loan amount, loan maturity, guarantors and availability of collateral.

The decision-making is based on the information about the applicant addressing the following points: net income, age, last employment period, credit history, purpose of loan, requested loan amount, loan maturity, number of guarantors, and collateral. An excerpt from the data base including the input data items together with the corresponding expert decisions is presented in Table 3.

To develop an intelligent system to automatically evaluate loan applicants in terms of their creditability, the experts were interviewed and the initial rule-base was formed. To make the functionality of the decision-making process transparent and trustworthy, the inference system was constructed through a collection of fuzzy "If-Then" rules, which would keep it compact and human interpretable. The historical database collected during a certain period of activity of the bank (with information about 460 applicants) was used to validate and optimize the initial rule-base using the proposed clustering technique. In the rules, the entries (inputs) "Net income" (X1), "Age" (X2), "Last employment period" (X3), and "Requested loan amount" (X6) were treated as type-2 fuzzy input variables. The remaining entries were treated as type-1 fuzzy or numerical variables.

The derived rule-base consisted of fuzzy rules such as those presented below:

**Table 3**
An excerpt from loan applicants' database.

| # | Net income (USD) | Age (yrs) | Last empl. period (yrs) | Credit History | Purpose of loan | Requested loan amount | Loan maturity (yrs) | Number of guarantors | Collateral | Loan Request |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1073 | 29 | 3 | Negative | Flat refurb. | 3000 | 36 | 1 | N/A | Denied |
| 2 | 893 | 32 | 4 | Negative | Flat refurb. | 3000 | 36 | 2 | N/A | Denied |
| 3 | 664 | 25 | 2 | Positive | Car purch. | 6000 | 36 | 2 | Car | Accepted |
| 4 | 1348 | 34 | 2 | Positive | Car purch. | 8000 | 36 | 2 | Car | Accepted |
| 5 | 250 | 20 | 0.5 | Positive | Car purch. | 2000 | 24 | 2 | Car | Denied |
| 6 | 400 | 24 | 3 | Positive | Flat refurb. | 2500 | 12 | 1 | N/A | Accepted |
| 7 | 140 | 25 | 1 | Positive | Car purch. | 1500 | 30 | 2 | Car | Denied |
| 8 | 524 | 39 | 5 | Positive | Flat purch. | 5000 | 36 | 2 | Flat | Accepted |
| 9 | 662 | 32 | 4 | Positive | Flat purch. | 6500 | 36 | 1 | Flat | Accepted |
| 10 | 1695 | 37 | 7 | Positive | Flat purch. | 15,000 | 24 | 1 | Flat | Accepted |

**IF** "Net income" **IS** "High (X1)" **AND** "Age" **IS** "Average (X2)" **AND** "Last employment period" **IS** "High (X3)" **AND** "Credit history" **IS** "High (X4)" **AND** "Purpose of loan" **IS** "Flat purchase (X5)" **AND** "Requested loan amount" **IS** "Average (X6)" **AND** "Maturity of loan" **IS** "Average (X7)" **AND** "Number of guarantors" **IS** "At least one (X8)" **AND** "Collateral" **IS** "Don't care (X9)".

**THEN** Accept **IS** "Very high (Y)".

The rule-base was aligned with the clustering results and submitted to the experts for further revision. Finally, the approved rule-base contained 34 fuzzy "If-Then" rules. Variable term-sets were defined by initial membership functions which were afterwards adjusted by the training procedure realized in the suggested Type-2 Fuzzy Neural Network.

The performance of the obtained system was compared with the performance produced by the "standard" feed-forward neural network with 9 input (X1,...,X9), 10 neurons in the hidden layer (which was based the trial-and-error approach: the network with 10 neurons in hidden layer converged better than ones with 7–9 or more than 10 neurons), and 1 output neurons trained by means of the back-propagation learning scheme on available 460 data entries with an initial learning rate set to 0.6. Although both systems showed similar numeric results (performance), the advantage of the fuzzy system comes with respect to much higher value of trustworthiness of achieved decision due to the transparency of the underlying decision-making mechanism.

## 8. Concluding comments

In this study, we have discussed a concept of type-2 fuzzy inference system using type-2 fuzzy clustering and fuzzy neural-network-based refinements.

To express the uncertainty associated with the fuzzification parameter of the fuzzy clustering $m$, the type-2 fuzzy clustering problem was formulated. The underlying procedure for solving this problem was based on the use of the DE optimization. Further adjustments of the initial fuzzy rule-base formed through fuzzy clustering were performed by the Type-2 Fuzzy Inference Neural Network (T2FINN). The training of the fuzzy neural network was also realized by the DEO.

To offer a comprehensive performance analysis, several examples, including the benchmark problem of identification of nonlinear system and two applications, including oil production forecasting and loan assessment systems, were considered. The comparative analysis showed that type-2 fuzzy inference model outperforms the best of its counterparts and yields higher accuracy. In general, the type-2 based model has better performance than type-1 based model since the type-2 MF are characterized by more parameters than type-1 MFs and thus the type-2 inference system is able to deal with higher levels of uncertainty.

Further developments of the introduced concept of type-2 fuzzy clustering and neural network-based refinements may be related with further advancements of fuzzy clustering involving the use of a variety of distance functions, optimization of number of clusters, as the use of other classes of type-2 fuzzy sets.

## References


[1] F.S. Al-Anzi, A. Allahverdi, A self-adaptive differential evolution heuristic for two-stage assembly scheduling problem to minimize maximum lateness with setup times, European Journal of Operational Research 182 (2007) 80–94.

[2] R.A. Aliev, B. Fazlollahi, R.R. Aliev, Soft computing and its applications in business and economics, Springer, Series: Studies in Fuzziness and Soft Computing 157 (2004) 446. XVI.

[3] J. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum, New York, 1981.

[4] C. Borgelt, Accelerating fuzzy clustering, Information Sciences 179 (23) (2009) 3985–3997.

[5] J.R. Castro, O. Castillo, P. Melin, A. Rodríguez-Díaz, A hybrid learning algorithm for a class of interval type-2 fuzzy neural networks, Information Sciences 179 (13) (2009) 2175–2193.

[6] J. Chaneau, M. Gunaratne, A. Altschaeffl, J. Bezdek (Eds.), An application of type-2 fuzzy sets to decision making in engineering, in analysis of fuzzy information, Artificial Intelligence and Decision Systems, vol. 2, CRC, Boca Ration, FL, 1987.

[7] W.-D. Chang, Nonlinear system identification and control using a real-coded genetic algorithm, Applied Mathematical Modeling 31 (2007) 541–550.

[8] M.Y. Chen, D.A. Linkens, Rule-base self-generation and simplification for data-driven fuzzy models, Journal of Fuzzy Sets and Systems 142 (2004) 243–265.

[9] G. Folino, A. Forestiero, G. Spezzano, An adaptive flocking algorithm for performing approximate clustering, Information Sciences 179 (18) (2009) 3059–3078.

[10] H. Hagras, Comments on dynamical optimal training for interval type-2 fuzzy neural network (T2FNN), IEEE Transactions on Systems, Man, and Cybernetics-part B: Cybernetics 36 (5) (2006) 1206–1209.

[11] D. Hidalgo, O. Castillo, P. Melin, Type-1 and type-2 fuzzy inference systems as integration methods in modular neural networks for multimodal biometry and its optimization with genetic algorithms, Information Sciences 179 (13) (2009) 2123–2145.

[12] H. Huang, C. Wu, Approximation capabilities of multilayer fuzzy neural networks on the set of fuzzy-valued functions, Information Sciences 179 (16) (2009) 2762–2773.

[13] C. Hwang, F. Chung-Hoon Rhee, Uncertain fuzzy clustering: Interval Type-2 Fuzzy Approach to C-Means, IEEE, 2007. pp.107-120.

[14] P. Innocent, R. John, I. Belton, D. Finlay, Type 2 Fuzzy Representations of Lung Scans to Predict Pulmonary Emboli, in: Proceedings 2001 IFSA/NAFIPS Joint Conference, Vancouver, BC, Canada, 2001, pp. 1902–1907.

[15] R. John, Type-2 inferencing and community transport scheduling, in: Proceedings of Fourth European Congress on Intelligent Techniques Soft Computing, Aachen, Germany, September 1996, pp. 1369–1372.

[16] N.N. Karnik, J.M. Mendel, Introduction to type-2 fuzzy logic systems, in: Proceedings of IEEE FUZZ Conference, Anchorage, AK, May 1998.

[17] N.N. Karnik, J.M. Mendel, Operations on type-2 fuzzy sets, Fuzzy Sets and Systems (2000).

[18] N.N. Karnik, J.M. Mendel, Type-2 fuzzy logic systems, IEEE Transactions on Fuzzy Systems 7 (1999) 643–658.

[19] N.N. Karnik, J.M. Mendel, Type-2 fuzzy logic systems: type-reduction, in: IEEE Systems, Man, Cybernet Conference, San Diego, CA, October 1998.

[20] N.N. Karnik, J. Mendel, Application of type-2 fuzzy logic systems to forecasting of time series, Information Sciences 120 (1999) 89–111.

[21] Q. Liang, J.M. Mendel, Interval type-2 fuzzy logic systems: theory and design, IEEE Transactions on Fuzzy Systems 8 (2000) 535–550.

[22] R. Martínez, O. Castillo, L.T. Aguilar, Optimization of interval type-2 fuzzy logic controllers for a perturbed autonomous wheeled mobile robot using genetic algorithms, Information Sciences 179 (13) (2009) 2158–2174.
[23] J.M. Mendel, R.I. John, Type-2 fuzzy sets made simple, IEEE Transactions on Fuzzy Systems 10 (2002) 117–127.
[24] J.M. Mendel, Uncertain Rule-based Fuzzy Logic Systems: Introduction and New Directions, Prentice-Hall, 2001.
[25] G.M. Méndez, M. Hernandez, Hybrid learning for interval type-2 fuzzy logic systems based on orthogonal least-squares and back-propagation methods, Information Sciences 179 (13) (2009) 2146–2157.
[26] M. Mizumoto, K. Tanaka, Some properties of fuzzy sets of type 2, Information and Control 31 (1976) 312–340.
[27] I. Ozkan, I.B. Turksen, Upper and lower values for the level of fuzziness in FCM, Information Sciences 177 (23) (2007) 5143–5152.
[28] S. Park, H. Lee-Kwang, A Designing Method for Type-2 Fuzzy Logic Systems Using Genetic Algorithms, IEEE, 2001. pp. 2567–2572.
[29] W. Pedrycz, K. Hirota, Fuzzy vector quantization with particle swarm optimization: a study in fuzzy granulation-degranulation information processing, Signal Processing (2007), doi:10.1016/j.sigpro.2007.02.001.
[30] W. Pedrycz, Knowledge-Based Clustering: From Data to Information Granules, John Wiley & Sons, 2005. p. 316.
[31] K. Price, R. Storn, J. Lampinen, Differential Evolution – A Practical Approach to Global Optimization, Springer, Berlin, 2005.
[32] S. Saha, S. Bandyopadhyay, A new point symmetry based fuzzy genetic clustering technique for automatic evolution of clusters, Information Sciences 179 (19) (2009) 3230–3246.
[33] I.B. Türkşen, Non-specificity and interval valued fuzzy sets, Fuzzy Sets and Systems (Invited Special Issue), 1994.
[34] C.H. Wang, C.S. Cheng, T.T. Lee, Dynamical optimal training for interval type-2 fuzzy neural network (T2FNN), IEEE Transactions on Systems, Man, and Cybernetics-part B: Cybernetics 34 (3) (2004) 1462–1477.
[35] Y.-J. Wang, J.-S. Zhang, G.-Y. Zhang, A dynamic clustering based differential evolution algorithm for global optimization, European Journal of Operational Research 183 (2007) 56–73.
[36] K. Wu, Fuzzy interval control of mobile robots, Computers and Electrical Engineering 22 (1996) 211–229.
[37] R. Yager, Fuzzy subsets of type II in decisions, Journal of Cybernetics 10 (1980) 137–159.
[38] L.A. Zadeh, The concept of a linguistic variable and its application to approximate reasoning, Information Sciences 8 (1975) 199–249.