Chapter 2
Selected Topics in Fuzzy Systems Designing

The research field of fuzzy systems is based on the theory of fuzzy sets [78]. However, it comprises a number of issues, which are related to, among others, triangular norms (see e.g. [2, 30, 31]), negations (see e.g. [59, 68]), inference operators (see e.g. [4, 5, 39, 44]), defuzzification methods (see e.g. [18, 34, 73]), a way of learning (see e.g. [54–56]), etc. For this reason, the scope of this chapter is limited and it describes the issues necessary for proper understanding of the contents presented in the subsequent chapters. The chapter contains a description of Mamdani-type and logical-type fuzzy systems (Sect. 2.1), an introduction to the learning methods of fuzzy systems (Sect. 2.2), and the evaluation methods of their performance when used in different application areas (Sect. 2.3).

2.1 Fuzzy System Description

This section provides a description of the fuzzy system which is used to present aspects of interpretable fuzzy systems designing. This system may have multiple inputs and multiple outputs (after the generalization of one-output system). Thus, it maps $\mathbf{X} \rightarrow \mathbf{Y}$, where input space data $\mathbf{X} = X_1, \ldots, X_n \subset \mathbf{R}^n$ and output space data $\mathbf{Y} = Y_1, \ldots, Y_m \subset \mathbf{R}^m$. The structure of the considered fuzzy system includes four main blocks: fuzzy rules base, inference block, fuzzification block and defuzzification block.

Fuzzification block performs a transformation (fuzzification operation) of not fuzzy (sharp) space $\mathbf{X} \subset \mathbf{R}^n$ to the space of fuzzy sets defined in $\mathbf{X}$. Due to fuzzification it is possible to give real number values (typical for most practical applications) at the system inputs. The most often realized fuzzification operation is so-called singleton fuzzification, which maps $\bar{\mathbf{x}} = [\bar{x}_1, \ldots, \bar{x}_n] \in \mathbf{X}$ to the fuzzy set $A' \subseteq \mathbf{X}$ which has the following membership function:

K. Cpałka, *Design of Interpretable Fuzzy Systems*, Studies in Computational Intelligence 684, DOI 10.1007/978-3-319-52881-6_2

$$\mu_{A'}(\mathbf{x}) = \begin{cases} 1 \text{ if } \mathbf{x} = \bar{\mathbf{x}} \\ 0 \text{ if } \mathbf{x} \neq \bar{\mathbf{x}} \end{cases}. \tag{2.1}$$

In addition to singleton fuzzification, there is also non-singleton fuzzification (see e.g. [27, 41, 54]). It can be used e.g. for filtering noise from input signals of a fuzzy system.

A fuzzy rules base contains the collection of N fuzzy rules. Each of them can be interpreted as a fuzzy relation on the set $\mathbf{X} \times \mathbf{Y}$. It has the following form:

$$R^k : \begin{bmatrix} \text{IF } \left(x_1 \text{ is } A_1^k\right) \text{ AND} \ldots \text{AND } \left(x_n \text{ is } A_n^k\right) \\ \text{THEN } \left(y_1 \text{ is } B_1^k\right), \ldots, \left(y_m \text{ is } B_m^k\right) \end{bmatrix}, \tag{2.2}$$

where n is the number of input linguistic variables (system inputs), m is the number of output linguistic variables (system outputs), $\mathbf{x} = [x_1, \ldots, x_n] \in \mathbf{X}$ is the vector of input linguistic variables, $\mathbf{y} = [y_1, \ldots, y_m] \in \mathbf{Y}$ is the vector of output linguistic variables, $A_1^k, \ldots, A_n^k$ $(k = 1, \ldots, N)$ are input fuzzy sets representing values of input linguistic variables, $B_1^k, \ldots, B_m^k$ $(k = 1, \ldots, N)$ are output fuzzy sets representing values of output linguistic variables, $\mu_{A_i^k}(x_i)$ are membership functions of input fuzzy sets and $\mu_{B_j^k}\left(y_j\right)$ are membership functions of output fuzzy sets. Thus, linguistic variables do not have numeric values but the ones described by fuzzy sets. These sets represent descriptive notions like "low", "high", "close to the value of 5", etc. The notions cannot be directly processed e.g. in artificial neural-networks. Shape and distribution analysis of input and output fuzzy sets has an important impact on their interpretability.

The inference block processes input fuzzy values of the system and generates the output fuzzy value. First, fuzzy conclusions are determined from the system rules. They have a form of fuzzy sets $\bar{B}_j^k$, generated independently for each rule k and each output j of the system. When using the modus ponens generalized inference rule, which is the most commonly used [56], the set $\bar{B}_j^k$ has the following form:

$$\bar{B}_j^k = A' \circ \left(\mathbf{A}^k \rightarrow B_j^k\right), \tag{2.3}$$

where $\mathbf{A}^k \rightarrow B_j^k$ is the fuzzy relation represented by the fuzzy rule R^k and $\mathbf{A}^k = A_1^k \times \ldots \times A_n^k$ is a Cartesian product of fuzzy sets $A_1^k, \ldots, A_n^k$. Values of membership functions of fuzzy sets $\bar{B}_j^k$ of the form (2.3) are determined as follows:

$$\mu_{\bar{B}_j^k}\left(y_j\right) = \sup_{\mathbf{x} \in \mathbf{X}} \left\{ T \left\{ \mu_{A'}(\mathbf{x}), \mu_{\mathbf{A}^k \rightarrow B_j^k}\left(\mathbf{x}, y_j\right) \right\} \right\}, \tag{2.4}$$

where t-norm $T\{\cdot\}$ is a generalization of a conjunction operator known from a bivalent logic (see e.g. [2, 30, 31, 55, 56]). A set of sample t-norms is presented in Table 2.1. Using the singleton defuzzification (2.1) and a boundary condition for t-norm, the dependency (2.4) can be simplified to the following form:

Table 2.1 A list of exemplary triangular norms

Name	t-norm and t-conorm
Minimum/Maximum	$\begin{cases} T_m\{\mathbf{a}\} = \min\limits_{i=1,\ldots,n}\{a_i\} \\ S_m\{\mathbf{a}\} = \max\limits_{i=1,\ldots,n}\{a_i\} \end{cases}$
Łukasiewicz	$\begin{cases} T_L\{\mathbf{a}\} = \max\left\{\sum\limits_{i=1}^{n} a_i - (n-1), 0\right\} \\ S_L\{\mathbf{a}\} = \min\left\{\sum\limits_{i=1}^{n} a_i, 1\right\} \end{cases}$
Algebraic	$\begin{cases} T_a(\mathbf{a}) = \prod\limits_{i=1}^{n} a_i \\ S_a(\mathbf{a}) = 1 - \prod\limits_{i=1}^{n}(1-a_i) \end{cases}$
Drastic	$\begin{cases} T_d\{\mathbf{a}\} = \begin{cases} 0 & \text{for } S_m\{\mathbf{a}\} < 1 \\ T_m\{\mathbf{a}\} & \text{for } S_m\{\mathbf{a}\} = 1 \end{cases} \\ S_d\{\mathbf{a}\} = \begin{cases} 1 & \text{for } T_m\{\mathbf{a}\} > 0 \\ S_m\{\mathbf{a}\} & \text{for } T_m\{\mathbf{a}\} = 0 \end{cases} \end{cases}$

$$\mu_{\bar{B}_j^k}(y_j) = \mu_{\mathbf{A}^k} \rightarrow B_j^k(\bar{\mathbf{x}}, y_j) = I\left(\mu_{\mathbf{A}^k}(\bar{\mathbf{x}}), \mu_{B_j^k}(y_j)\right), \tag{2.5}$$

where $I(\cdot)$ is an inference operator depending on the inference type (see Sects. 2.1.1 and 2.1.2). Influencing the accuracy of inference operator used in Eq. (2.5) is an important aspect of influence on interpretability. Notation $\mathbf{A}^k$ in dependency (2.5) is interpreted as the activity level of the rule R^k and it is determined as follows:

$$\mu_{\mathbf{A}^k}(\bar{\mathbf{x}}) = T\left\{\mu_{A_1^k}(\bar{x}_1), \ldots, \mu_{A_n^k}(\bar{x}_n)\right\} = \underset{i=1}{\overset{n}{T}}\left\{\mu_{A_i^k}(\bar{x}_i)\right\} = \tau_k(\bar{\mathbf{x}}). \tag{2.6}$$

Analysis of rules activity and influencing a precision of the aggregation operator (t-norm) used in Eq. (2.6) are an important aspect of influence on interpretability.

Next, conclusions from fuzzy rules $\bar{B}_j^k$ (their number is equal to $N \cdot m$) generated in the inference block are aggregated to the fuzzy conclusions from the whole rule base (their number is equal to m). They take the form of fuzzy sets B_j' $(j = 1, \ldots, m)$. The sets are described by membership functions $\mu_{B_j'}(y_j)$. They are created as a result of aggregation of the sets $\bar{B}_j^k$ in a manner dependent on the assumed inference type (Sects. 2.1.1 and 2.1.2). In addition to the described implementation of the inference block, there are also other ways based e.g. on another inference rule (e.g. the modus tollens generalized inference rule), another way of generating conclusions from the fuzzy rules base, etc. (see e.g. [56]).

The defuzzification block performs the inverse operation of the one implemented in the fuzzification block. It provides a transition from the fuzzy sets B_j'

($j = 1, \ldots, m$) obtained on the output of the inference block to the real (sharp) output signals $\bar{y}_j$ from the fuzzy system in the space $\mathbf{Y} \subset \mathbf{R}$. In practice, there are many different defuzzification operators (see e.g. [18, 34, 73]) and new ones are still being developed. For the purpose of further consideration we have assumed that the inference will be implemented by the commonly used center-of-area method. It has the following form:

$$\bar{y}_j = \frac{\int\limits_{\mathbf{Y}} y_j \cdot \mu_{B'_j}\left(y_j\right) dy_j}{\int\limits_{\mathbf{Y}} \mu_{B'_j}\left(y_j\right) dy_j}. \tag{2.7}$$

The discrete form of dependency (2.7) is usually written as follows:

$$\bar{y}_j = \frac{\sum\limits_{r=1}^{N} \bar{y}_{j,r}^{B} \cdot \mu_{B'_j}\left(\bar{y}_{j,r}^{B}\right)}{\sum\limits_{r=1}^{N} \mu_{B'_j}\left(\bar{y}_{j,r}^{B}\right)}, \tag{2.8}$$

where $\bar{y}_{j,r}^{B}$ ($r = 1, \ldots, N$, $j = 1, \ldots, m$) are discretization points of fuzzy set B'_j. Most often they are the points at which input fuzzy sets B_j^k reach their maximum values. If fuzzy sets B_j^k reach their maximum in a range value, then, another defuzzification method can be used. However, it should be noted that most of the defuzzification methods is dependent on the number N of the system rules. It means a negative dependency between precision of defuzzification operator and the number of fuzzy rules. An analysis of this issue is an important aspect of affecting interpretability.

2.1.1 *Fuzzy System Implementing Mamdani-Type Inference*

Specifics of the Mamdani-type fuzzy system result from an inference block implementation method. In this system a t-norm is most often the inference operator. However, attempts are also being made to use other operators which meet some or all assumptions arising from the t-norm definition (see e.g. [31, 38, 44]). In the Mamdani-type system the dependency (2.5) can be written as follows:

$$I\left(\mu_{\mathbf{A}^k}\left(\bar{\mathbf{x}}\right), \mu_{B_j^k}\left(y_j\right)\right) = T\left\{\mu_{\mathbf{A}^k}\left(\bar{\mathbf{x}}\right), \mu_{B_j^k}\left(y_j\right)\right\}. \tag{2.9}$$

In turn, aggregation of fuzzy conclusions $\bar{B}_j^k$ from fuzzy rules R^k to final conclusions B'_j ($j = 1, \ldots, m$) is performed as follows:

$$B'_j = \bigcup_{k=1}^{N} \bar{B}_j^k. \tag{2.10}$$

Membership functions of fuzzy sets B'_j are determined using the following formula:

$$\mu_{B'_j}(y_j) = S\left\{\mu_{\bar{B}^1_j}(y_j), \ldots, \mu_{\bar{B}^N_j}(y_j)\right\} = \underset{k=1}{\overset{N}{S}}\left\{\mu_{\bar{B}^k_j}(y_j)\right\}, \tag{2.11}$$

where t-conorm $S\{\cdot\}$ is a generalization of an alternative operator known from a bivalent logic (see e.g. [2, 30, 31, 55, 56]). It means that the rules of the form (2.2) are associated by the operator which is an extension of the OR operator. A set of sample t-conorms is presented in Table 2.1.

Taking into account formulas (2.6), (2.9) and (2.11) in the assumed defuzzification formula (2.8), a dependency describing the value of the output signal $\bar{y}_j$ for the Mamdani-type system is obtained:

$$\bar{y}_j = \frac{\sum\limits_{r=1}^{N} \bar{y}^B_{j,r} \cdot \underset{k=1}{\overset{N}{S}}\left\{T\left\{\underset{i=1}{\overset{n}{T}}\left\{\mu_{A^k_i}(\bar{x}_i)\right\}, \mu_{B^k_j}\left(\bar{y}^B_{j,r}\right)\right\}\right\}}{\sum\limits_{r=1}^{N} \underset{k=1}{\overset{N}{S}}\left\{T\left\{\underset{i=1}{\overset{n}{T}}\left\{\mu_{A^k_i}(\bar{x}_i)\right\}, \mu_{B^k_j}\left(\bar{y}^B_{j,r}\right)\right\}\right\}}. \tag{2.12}$$

A system implementing an alternative inference type is a logical-type system. It is described in the next section.

2.1.2 *Fuzzy System Implementing Logical-Type Inference*

Specifics of the logical-type fuzzy system also (like in the Mamdani-type system) result from an inference block implementation method. In this system the inference operator may be e.g.: s-implication, r-implication, q-implication, or other operators which are usually an extension of logical implication (see e.g. [4, 5, 31, 38, 44]). In this book systems using s-implication inference operator are considered. Then, the dependency (2.5) can be presented as follows:

$$I\left(\mu_{\mathbf{A}^k}(\bar{\mathbf{x}}), \mu_{B^k_j}(y_j)\right) = S\left\{\text{neg}\left(\mu_{\mathbf{A}^k}(\bar{\mathbf{x}})\right), \mu_{B^k_j}(y_j)\right\}, \tag{2.13}$$

where neg $(\cdot)$ is a negation operator (see e.g. [31, 44, 59, 68]). There are many different operators consistent with the definition of the negation (e.g. Zadeh, Hamacher, Sugeno, Dombi, Yager, etc.), but in practice Zadeh negation is actually the one which is most often used:

$$\text{neg}\left(\mu_{\mathbf{A}^k}(\bar{\mathbf{x}})\right) = 1 - \mu_{\mathbf{A}^k}(\bar{\mathbf{x}}). \tag{2.14}$$

In the considered system of the logical-type an aggregation of fuzzy conclusions $\bar{B}^k_j$ from fuzzy rules R^k to final conclusions B'_j $(j = 1, \ldots, m)$ is performed as follows:

$$B'_j = \bigcap_{k=1}^{N} \bar{B}_j^k. \tag{2.15}$$

Membership functions of fuzzy sets B'_j are determined using the following formula:

$$\mu_{B'_j}(y_j) = T\left\{\mu_{\bar{B}_j^1}(y_j), \ldots, \mu_{\bar{B}_j^N}(y_j)\right\} = \underset{k=1}{\overset{N}{T}}\left\{\mu_{\bar{B}_j^k}(y_j)\right\}. \tag{2.16}$$

It means that the rules of the form (2.2) are associated by the operator which is an extension of the AND operator.

Taking into account formulas (2.6), (2.13), (2.14) and (2.16) in the assumed defuzzification formula (2.8), a dependency describing the value of the output signal $\bar{y}_j$ for the logical-type system is obtained:

$$\bar{y}_j = \frac{\sum_{r=1}^{N} \bar{y}_{j,r}^B \cdot \underset{k=1}{\overset{N}{T}}\left\{S\left\{1 - \underset{i=1}{\overset{n}{T}}\left\{\mu_{A_i^k}(\bar{x}_i)\right\}, \mu_{B_j^k}\left(\bar{y}_{j,r}^B\right)\right\}\right\}}{\sum_{r=1}^{N} \underset{k=1}{\overset{N}{T}}\left\{S\left\{1 - \underset{i=1}{\overset{n}{T}}\left\{\mu_{A_i^k}(\bar{x}_i)\right\}, \mu_{B_j^k}\left(\bar{y}_{j,r}^B\right)\right\}\right\}}. \tag{2.17}$$

Systems of the forms (2.12) and (2.17) are the base systems for the considerations presented in Chap. 4.

2.2 Fuzzy System Learning Methods

In practice, there are several common approaches to determining the structure and parameters of fuzzy systems:

- The first one assumes that the fuzzy rules and their fuzzy sets are formulated by an expert. It determines the structure and parameters of a system. If indications of the expert are accurate, then the learning of such fuzzy system is not needed.
- The second approach assumes that we have the expert knowledge (as it is in the first approach) and a learning sequence containing learning sets. Each set contains input signals of the system and corresponding output reference signals. Then, the system initialized by the expert can be also learned (tuned), thus increasing the precision of its operation.
- The third approach (the most common) assumes that we have only learning data at our disposal. They are the basis for learning of the fuzzy system. The learning may be unsupervised or/and supervised (see e.g. [47, 48]). In the most basic applications the purpose of learning is selection of parameters of the fuzzy system whose structure has been specified by the designer (e.g. the number of rules determined by trial and error). The purpose of learning is also increasingly more often an automatic selection of the fuzzy system structure taking into account, among

others, additional requirements. They may include e.g. the degree of complexity, aspects of readability, etc. This approach is considered in Chaps. 4–7.

- The fourth approach also assumes that we only have learning data at our disposal. However, they are not used for iterative learning of the system. They are used in order to generate appropriate descriptors, which have a precise interpretation (purpose) in the fuzzy system. This approach is considered in Chap. 8.

As already mentioned, the issue of fuzzy systems learning is based on capabilities of unsupervised and supervised learning. Unsupervised learning can use data clustering techniques (see e.g. [1, 20, 72]). They are one of the oldest and most popular data mining methods. Their aim is to extract classes in a data set and associate them with objects on the basis of an assumed similarity function. Use of data clustering in fuzzy systems learning allows us, among others, to properly initiate parameters of the system and outline its structure (e.g. the number of fuzzy rules). Here the use of a cluster validity index is often found helpful. An example of using unsupervised learning for selection (initialization) of the fuzzy system structure is shown in Chap. 5.

However, the learning of fuzzy systems is mainly performed using supervised learning, which is based on two types of algorithms:

- Gradient algorithms. The purpose of the gradient algorithm is iterative selection of the direction of optimized objective function changes (in particular, the evaluation function of the fuzzy system) in order to find its extreme (usually the minimum). Determination of the direction of change for the next step is based on the knowledge of the optimized objective function gradient in the current step. The direction of change being sought is the one in which the optimized objective function changes with the greatest intensity (decreases or increases). The most common gradient methods used in practice include the error backpropagation method [11, 45, 69, 70], its variant with the so-called momentum term [52, 76, 77], the Levenberg–Marquardt method [33, 37, 75], least squares method [67], the Fletcher-Reeves conjugate gradient method [22], sub-gradient methods [6], etc.
- Population algorithms. These algorithms are heuristic methods and they process a population of solutions in order to effectively search a given space of considerations (which takes into account the range of searched parameters). A single solution of a population is called an individual or a chromosome (in the case of genetic algorithms). Such individual may be e.g. a suitably encoded system described by formula (2.8). It is evaluated by the evaluation function (fitness function). Its form is dependent on a given problem. The value of the evaluation function of an individual determines its chances of survival in a given population. In the case of fuzzy systems, not only can the accuracy of the system be evaluated, but also e.g. the complexity of its structure and readability of its rules.

At present, the topics related to population algorithms are developing intensively. The reasons for that include high flexibility of this class of methods in configuration of a population and also ways of its processing, which can result in various approaches to

their division. Therefore, taking into account the issue of configuration of population, these algorithms can be divided into:

- Single-population algorithms. They use a single population of individuals. A generalized scheme of operation of such algorithms includes the following steps: (a) Initiation of individuals in a population. (b) Evaluation of individuals in the given population using a defined fitness function. (c) Modification of individuals according to the specifics of the algorithm. (d) Evaluation of new individuals in the population. (e) Replacing the old population with the new population of individuals according to the specifics of the algorithm (this step can be omitted for those algorithms which do not create new populations, but only modify the parameters of an existing population). (f) Checking the stopping criterion, which may take into account e.g. the execution of a certain number of steps or getting a satisfactory solution in the form of an adopted evaluation function. In the case where this condition is not met, the algorithm returns to step c. However, if the stop condition is satisfied, then the presentation of the best individual is performed and the algorithm terminates. A single sequence of steps c–f is called a single iteration of the algorithm or a single evolution.
- Multi-population algorithms. Operation scheme of multi-population algorithms is analogous to the single-population algorithms. The difference results from the need to manage multiple populations. Each component population can be learned individually. Later those populations can either compete or cooperate. In Table 2.2 typical interactions between populations determining specificity of the algorithm are shown. In multi-population algorithms there is also the phenomenon of migration, which is the transfer of individuals between component populations. The way of migration can be determined by the result of competition. Common migration topologies are shown in Fig. 2.1. The use of many populations is implemented so

Table 2.2 Summary of possible interactions that can occur between populations

Type of interaction	Impact on		Description of relation
	Population 1	Population 2	
Neutralism	None	None	Populations are independent
Mutualism	Positive	Positive	Mutual benefit
Commensalism	Positive	None	One-sided benefit
Competition	Negative	Negative	Both sides are negatively affected
Predation	Positive	Negative	One-sided benefits, while the other is negatively affected
Parasitism	Positive	Negative	One-sided benefits, while the other is negatively affected

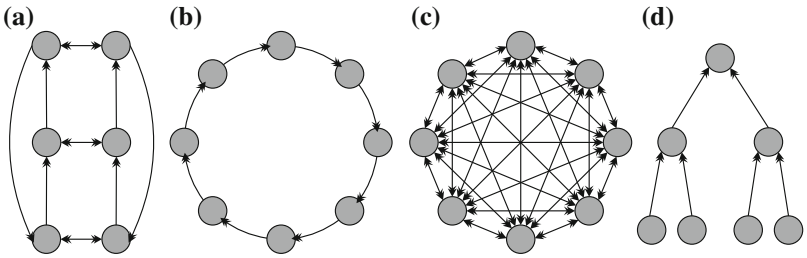

Fig. 2.1 Typical migration topologies between populations (the *arrows* indicate direction of migration): **a** ladder-type topology [9], **b** one-direction circle topology [8], **c** peer-to-peer topology [8], **d** hierarchical topology [49]

as to increase diversity of individuals in component populations and to improve the resistance of the algorithm to (negative) search for optimal solutions only locally.

Another criterion for division of population algorithms can be the way how a population is processed. In this context, population algorithms may be categorized, among others, as follows:

- Evolutionary algorithms. This group of algorithms can include all population-based algorithms which use the processing of population individuals based on biological evolution. The most commonly used evolutionary algorithms include genetic algorithms [24], evolution strategies [7], genetic programming [14, 51], evolutionary programming [23], differential evolution algorithms [21, 62–64], multi-population genetic algorithms [42], etc.
- Swarm algorithms. This group of algorithms may include all population-based algorithms which use processing of population individuals based on swarm behavior. The most commonly used swarm algorithms comprise the particle swarm optimization algorithm [12, 50], the artificial bee colony algorithm [29, 74], the intelligent water drops algorithm [60], the artificial immune algorithm [17, 65], the multi-swarm cooperative particle swarm optimization algorithm [46], etc.
- Other population-based algorithms. This group of algorithms can include all population-based algorithms which are difficult to define as evolutionary or swarm ones. They are algorithms based on league functioning (e.g. the golden ball algorithm [49]), biogeography algorithms (e.g. biogeography-based optimization algorithm [61]), algorithms based on colonization (e.g. invasive weed optimization [35]), algorithms referring to ways of spreading sparks (e.g. fireworks optimization algorithm [66]), algorithms modelling social revolution (e.g. multi-population imperialist competitive algorithm [3]), algorithms based on symbiosis (e.g. Particle Swarm^2 Optimization, PS^{2}0 [32]), etc.

From a practical point of view population algorithms can be grouped in a number of ways, but this, however, is not essential from the point of view of interpretable fuzzy systems design issues.

2.3 Methods for Evaluation of Fuzzy System Accuracy

Evaluation of fuzzy system accuracy depends on the field of application and formulated requirements (problem domain). It is implemented differently in classification issues and in regression (modelling) issues. Later in this chapter we describe basic ways used to evaluate performance of fuzzy systems used for classification and modelling. Moreover, it is pointed out that assessment of fuzzy systems accuracy in the issue of designing interpretable fuzzy systems is only one of the required/necessary/most often applied criteria for their overall evaluation.

2.3.1 Accuracy Evaluation of the Systems Used for Modelling

Accuracy evaluation of the systems used for modelling is most often performed using the root mean squared error (RMSE), which is defined as follows:

$$\text{RMSE}\,(\mathbf{fs}) = \frac{1}{m} \cdot \sum_{j=1}^{m} \sqrt{\frac{1}{Z} \cdot \sum_{z=1}^{Z} \left(d_{z,j}^{L} - \bar{y}_{z,j} \right)^2}, \tag{2.18}$$

where **fs** represents any fuzzy system being evaluated (especially a system described by Eq. (2.8)), m is the number of the system outputs, Z is the number of samples from a learning or testing sequence, $d_{z,j}^{L}$ is an expected output value for output j ($j = 1, \ldots, m$) for input vector z ($z = 1, \ldots, Z$), $\bar{y}_{z,j}$ is a real output value j ($j = 1, \ldots, m$) computed by the system for input vector z ($z = 1, \ldots, Z$).

Since formula (2.18) does not take into account the disparity between the values of different outputs of a multiple input multiple output (MIMO) system, in this book a normalized error measure is also used and it has the following form:

$$\text{acc}\,(\mathbf{fs}) = \frac{1}{m} \cdot \sum_{j=1}^{m} \frac{\frac{1}{Z} \cdot \sum_{z=1}^{Z} \left| d_{z,j}^{L} - \bar{y}_{z,j} \right|}{\max\limits_{z=1,\ldots,Z} \left\{ d_{z,j}^{L} \right\} - \min\limits_{z=1,\ldots,Z} \left\{ d_{z,j}^{L} \right\}}. \tag{2.19}$$

2.3.2 Accuracy Evaluation of the Systems Used for Classification

Fuzzy systems used for classification are characterized by various factors including the number of outputs, which is equal to the number of considered classes. In this case the input set belongs to a class associated with the system output whose signal value

is the highest. Therefore, accuracy evaluation of the systems used for classification is performed using a standard classification error, which is determined as follows:

$$\text{acc}\,(\mathbf{fs}) = 100\% \cdot \frac{1}{Z} \sum_{z=1}^{Z} \begin{cases} 1 \text{ for } yi_z \neq di_z \\ 0 \text{ for } yi_z = di_z \end{cases}, \tag{2.20}$$

where di_z is a reference index of the class for sample z ($z = 1, \ldots, Z$) and yi_z is an index of the class selected by the system for sample z ($z = 1, \ldots, Z$).

It should be noted that the 10-fold cross validation [71] is used most often for evaluation of the systems used for classification. In the 10-fold cross validation the learning sequence is divided into ten parts of equal size. Then, in the first step of this procedure the first part is treated as a testing sequence and the other 9 parts are treated as a learning sequence. In the next step of the procedure the second part becomes the testing sequence and the other ones act as the learning sequence, etc. Thus, the process is repeated ten times and the final result is determined by averaging the obtained partial results. In cross-validation procedure we often seek to preserve proportion between the number of class representatives which is similar to the one in the whole learning sequence. This is called stratified cross-validation.

2.3.3 *Accuracy Evaluation in the Context of Interpretable Fuzzy Systems Designing*

In interpretable fuzzy systems design issues system accuracy is only one of the evaluation criteria applied. The other criteria comprise complexity and interpretability. For this reason, solutions derived from multi-objective optimization [13, 15, 19, 25, 26, 36, 53] play an important role in the evaluation procedure. Multi-objective optimization allows us to take into account many components in the evaluation function. As a result, it is well suited for the evaluation of fuzzy systems in the context of, among others, accuracy, complexity and readability of rules. When considering the criteria for fuzzy system evaluation we may also affect their proper hierarchy of importance. Multi-criteria optimization methods also include:

- Scalar methods. They rely on aggregation of multiple criteria and require a priori knowledge. They include aggregation methods (e.g. random distribution of weights within a population [28]), methods based on LP-metrics [57], methods of ε-constraint type (i.e. a set of methods imposing restrictions on each evaluation function component [40]) as well as purpose methods (goal programming), in which the values of evaluation function components are optimized to certain threshold values [10].
- Methods based on the Pareto front. The front is a set of non-dominated solutions. They are the solutions for which there is no other better solution in terms of any criterion. These methods include ranking methods (based on a ranking of solutions,

e.g. NSGA-II [16]), elite methods (based only on non-dominated solutions, e.g. SEEA [80]) and methods managing the division (diversity maintaining [79]).
- Other methods. They include, among others, multi-population methods, methods based on interactions between populations (e.g. ERMOCS [43]) and methods with so-called parallel selection, which create vectors ordering sets of solutions in respect to each criterion independently (e.g. VEGA [58]).

Multi-criteria optimization techniques are perfectly suited to be used with population-based algorithms in the context of interpretable fuzzy system designing. These solutions are considered in Chap. 6.

2.4 Summary

In this chapter some selected issues in the field of fuzzy systems are outlined. Out of necessity the chapter focuses on the issues which are crucial for interpretable fuzzy systems designing. In particular, two types of fuzzy systems are described, i.e. the system implementing Mamdani-type inference and the system implementing logical-type inference. It is also emphasized that the considerations presented in this book have a general nature and they are not only related to the described class of systems. The presented description indicates the issues which have particular relevance to the subject matter of this book. Further on the chapter contains a review of basic techniques of fuzzy systems learning and methods for their evaluation. It discusses the aspects of supervised and unsupervised learning in the designing of fuzzy systems. Moreover, it emphasizes the importance of association of multi-objective optimization and evolutionary algorithms. The final part of the chapter indicates that accuracy of fuzzy systems is only one of the possible criteria for their evaluation.

References

1. Aggarwal, C.C., Reddy, C.K.: Data Clustering: Algorithms and Applications. Chapman and Hall/CRC, New York (2013)
2. Alsina, C., Maurice, F., Schweizer, B.: Associative Functions: Triangular Norms and Copulas. WSPC, New Jersey (2006)
3. Atashpaz–Gargari, E., Lucas, C.: Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. In: Proceedings of the IEEE Congress on Evolutionary Computation, vol. 7, pp. 4661–4666 (2007)
4. Baczyński, M.: Fuzzy Implications. Springer, Heidelberg (2008)
5. Baczyński, M., Beliakov, G., Sola, H.B., Pradera, A. (eds.): Advances in Fuzzy Implication Functions. Springer, New York (2015)
6. Bertsekas, D.P.: Incremental proximal methods for large scale convex optimization. Math. Progr. **129**, 163–195 (2011)
7. Beyer, H.G., Schwefel, H.P.: Evolution strategies: a comprehensive introduction. Nat. Comput. **1**, 3–52 (2002)

8. Borovska, P., Lazarova, M.: Migration policies for Island genetic models on multicomputer platform. In: Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS 2007), pp. 143–148 (2007)
9. Cantu-Paz, E.: Topologies, migration rates, and multi-population parallel genetic algorithms. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 1999), pp. 91–98 (1999)
10. Charnes, A., Cooper, W.W., Ferguson, R.: Optimal estimation of executive compensation by linear programming. Manag. Sci. **1**, 138–151 (1955)
11. Chauvin, Y., Rumelhart, D.E. (eds.): Backpropagation: Theory, Architectures, and Applications. Psychology Press (2013)
12. Clerc, M.: Particle Swarm Optimization. Wiley-ISTE, London (2006)
13. Collette, Y., Siarry, P.: Multiobjective Optimization: Principles and Case Studies. Springer, New York (2011)
14. Cramer, N.L.: A representation for the adaptive generation of simple sequential programs. In: Proceedings of the International Conference on Genetic Algorithms and their Applications, pp. 183–187 (1985)
15. Deb, K.: Multi-Objective Optimization Using Evolutionary Algorithms. Wiley, New York (2009)
16. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**, 182–197 (2002)
17. De Castro, L.N.: Artificial Immune Systems: A New Computational Intelligence Approach. Springer, New York (2002)
18. De Oliveira, J.V.: A set-theoretical defuzzification method. Fuzzy Sets Syst. **76**, 63–71 (1995)
19. Donoso, Y., Fabregat, R.: Multi-Objective Optimization in Computer Networks Using Metaheuristics. Auerbach Publications, Boca Raton (2007)
20. Everitt, B.S., Landau, S., Leese, M., Stahl, D.: Cluster Analysis. Wiley, New York (2011)
21. Feoktistov, V.: Differential Evolution: In Search of Solutions. Springer, New York (2006)
22. Fletcher, R., Reeves, C.: Function minimization by conjugate gradients. Comput. J. **7**, 149–154 (1964)
23. Fogel, L.J., Owens, A.J., Walsh, M.J.: Artificial Intelligence through Simulated Evolution. Wiley, New York (1966)
24. Fraser, A., Burnell, D.: Computer Models in Genetics. McGraw-Hill, New York (1970)
25. Gorzałczany, M., Rudziński, F.: A multi-objective genetic optimization for fast, fuzzy rule-based credit classification with balanced accuracy and interpretability. Appl. Soft Comput. **40**, 206–220 (2016)
26. Gorzałczany, M.B., Rudziński, F.: Accuracy versus interpretability of fuzzy rule-based classifiers: an evolutionary approach. In: Proceedings of the International Conference on Swarm and Evolutionary Computation (SIDE'12), pp. 222–230 (2012)
27. Heng, S., Chen, W., Yin, H., Shiping, M., Jizhang, Z.: Decision feedback equalizer based on non-singleton fuzzy regular neural networks. J. Syst. Eng. Electron. **17**, 896–900 (2006)
28. Jin, Y., Okabe, T., Sendhoff, B.: Adapting weighted aggregation for multiobjective evolutionary strategies. LNCS 1993, vol. 1993, pp. 96–110. Springer, Heidelberg (2006)
29. Karaboga, D.: An idea based on honey bee swarm for numerical optimization. Technical report-tr06. Erciyes University (2005)
30. Klement, E.P., Mesiar, R. (eds.): Logical, Algebraic, Analytic and Probabilistic Aspects of Triangular Norms. Elsevier, Amsterdam (2005)
31. Klement, E.P., Mesiar, R., Pap, E.: Triangular Norms. Springer, New York (2000)
32. Lenin, M.K., Reddy, B.R.: PS2O, hybrid evolutionary-conventional algorithm, genetical swarm optimization for solving reactive power optimization problem. Int. J. Recent Res. Electr. Electron. Eng. (IJRREEE) **1**, 10–20 (2014)
33. Levenberg, K.: A method for the solution of certain non-linear problems in least squares. Quart. Appl. Math. **2**, 164–168 (1944)
34. Mahdiani, H.R., Banaiyan, A., Javadi, M.H.S., Fakhraie, S.M., Lucas, C.: Defuzzification block: new algorithms, and efficient hardware and software implementation issues. Eng. Appl. Artif. Intell. **26**, 162–172 (2013)

35. Mallahzadeh, A.R.R., Oraizi, H., Davoodi–Rad, Z.: Application of the invasive weed optimization technique for antenna configurations. Prog. Electromagn. Res. **79**, 137–150 (2008)
36. Marler, T.: Multi-Objective Optimization: Concepts and Methods for Engineering. VDM Verlag, Germany (2009)
37. Marquardt, D.W.: An algorithm for least-squares estimation of nonlinear parameters. J. Soc. Ind. Appl. Math. **11**, 431–441 (1963)
38. Mas, M., Monserrat, M., Ruiz-Aguilera, D., Torrens, J.: RU and (U, N)-implications satisfying Modus Ponens. Int. J. Approx. Reason. (2016). doi:10.1016/j.ijar.2016.01.003
39. Mas, M., Monserrat, M., Torrens, J., Trillas, E.: A survey on fuzzy implication functions. IEEE Trans. Fuzzy Syst. **15**, 1107–1121 (2007)
40. Mavrotas, G.: Effective implementation of the e-constraint method in multi-objective mathematical programming problems. Appl. Math. Comput. **213**, 455–465 (2009)
41. Mouzouris, G.C., Mendel, J.M.: Dynamic non-singleton fuzzy logic systems for nonlinear modeling. IEEE Trans. Fuzzy Syst. **5**, 199–208 (1997)
42. Murata, T., Ishibuchi, H.: MOGA: multi-objective genetic algorithms. In: proceedings of the IEEE International Conference on Evolutionary Computation, vol. 102, pp. 289–294 (1995)
43. Neef, M., Thierens, D., Arciszewski, H.: A case study of a multiobjective elitist recombinative genetic algorithm with coevolutionary sharing. In: Proceedings of the 1999 Congress on Evolutionary Computation, pp. 796–803 (1999)
44. Nguyen, H.T., Walker, E.A.: A First Course in Fuzzy Logic. Chapman and Hall/CRC, Boca Raton (2005)
45. Nikolaev, N., Iba, H.: Adaptive Learning of Polynomial Networks: Genetic Programming, Backpropagation and Bayesian Methods. Springer, Boston (2006)
46. Niu, B., Zhu, Y., He, X., Wu, H.: MCPSO: a multi-swarm cooperative particle swarm optimizer. Appl. Math. Comput. **185**, 1050–1062 (2007)
47. Okun, O.: Supervised and Unsupervised Ensemble Methods and their Applications. Springer, Heidelberg (2008)
48. Okun, O.: Applications of Supervised and Unsupervised Ensemble Methods. Springer, Heidelberg (2010)
49. Osaba, E.: Golden ball: a novel meta-heuristic to solve combinatorial optimization problems based on soccer concepts. Appl. Intell. **12**, 145–166 (2013)
50. Parsopoulos, K.E., Vrahatis, M.N.: Particle Swarm Optimization and Intelligence: Advances and Applications. IGI Global, Pennsylvania (2010)
51. Poli, R., Langdon, W.B., McPhee, N.F.: A Field Guide to Genetic Programming. Lulu Enterprises (2008)
52. Qiu, G., Varley, M.R., Terrell, T.J.: Accelerated training of backpropagation networks by using adaptive momentum step. Electron. Lett. **28**, 377–379 (1992)
53. Rudziński, F.: A multi-objective genetic optimization of interpretability-oriented fuzzy rule-based classifiers. Appl. Soft Comput. **38**, 118–133 (2016)
54. Rutkowska, D.: Neuro-Fuzzy Architectures and Hybrid Learning. Springer, Heidelberg (2002)
55. Rutkowski, L.: Flexible Neuro-Fuzzy Systems: Structures, Learning and Performance Evaluation. Springer, Heidelberg (2004)
56. Rutkowski, L.: Computational Intelligence. Springer, Heidelberg (2010)
57. Sadjadi, S.J., Habibian, M., Khaledi, V.: A multi-objective decision making approach for solving quadratic multiple response surface problems. Int. J. Contemp. Math. Sci. **3**, 1595–1606 (2008)
58. Schaffer, J.: Multiple objective optimization with vector evaluated genetic algorithms. In: Proceedings of the First International Conference on Genetic Algorithms, pp. 93–100 (1985)
59. Schneider, M., Shnaider, E., Kandel, A.: Applications of the negation operator in fuzzy production rules. Fuzzy Sets Syst. **34**, 293–299 (1990)
60. Shah-Hosseini, H.: The intelligent water drops algorithm: a nature-inspired swarm-based optimization algorithm. Int. J. Bio-Inspir. Comput. **1**, 71–79 (2009)
61. Simon, D.: Biogeography-based optimization. IEEE Trans. Evol. Comput. **12**, 702–713 (2008)
62. Simon, D.: Evolutionary Optimization Algorithms. Wiley, New Jersey (2013)

63. Słowik, A.: Application of geometric differential evolution algorithm to design minimal phase digital filters with a typical characteristics for their hardware or software implementation. In: Proceedings of the 12th International Conference on Artificial Intelligence and Soft Computing, vol. 7895, pp. 67–78 (2013)
64. Storn, R.: On the usage of differential evolution for function optimization. In: Proceedings of the Biennial Conference of the North American Fuzzy Information Processing Society (NAFIPS), pp. 519–523 (1996)
65. Tan, Y.: Artificial Immune System: Applications in Computer Security. IEEE Computer Society Press Systems Series, Wiley (2016)
66. Tan, Y., Zhu, Y.: Fireworks algorithm for optimization. In: Proceedings of the First International Advances in Swarm Intelligence (ICSI2010). LNCS, vol. 6145, pp. 355–364. Springer, Heidelberg (2010)
67. Teunissen, P.J.G.: Dynamic Data Processing: Recursive Least-squares. VSSD, Netherlands (2009)
68. Weber, S.: A general concept of fuzzy connectives, negations and implications based on t-norms and t-conorms. Fuzzy Sets Syst. **11**, 115–134 (1983)
69. Werbos, P.J.: Beyond regression: New tools for predictions and analysis in the behavioral sciences (Ph.D. thesis). Harvard University (1974)
70. Werbos, P.J.: The Roots of Backpropagation: From Ordered Derivatives to Neural Networks and Political Forecasting. Wiley, New York (1994)
71. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, Amsterdam (2005)
72. Xu, R., Wunsc, D.: Clustering. Wiley-IEEE Press, New York (2008)
73. Yager, R.R., Filev, D.: On the issue of defuzzification and selection based on a fuzzy set. Fuzzy Sets Syst. **55**, 255–271 (1993)
74. Yang, X.S., Cui, Z., Xiao, R., Gandomi, A.H., Karamanoglu, M.: Swarm Intelligence and Bio-Inspired Computation: Theory and Applications. Elsevier, Oxford (2013)
75. Yu, H., Wilamowski, B.M.: Levenberg–Marquardt training. The Industrial Electronics Handbook, Intelligent Systems **5**, (1963)
76. Yu, X.H., Chen, G.A., Cheng, S.X.: Acceleration of backpropagation learning using optimised learning rate and momentum. Electron. Lett. **29**, 1288–1290 (1993)
77. Yu, H.H., Chen, G.A., Cheng, S.X.: Dynamic learning rate optimization of the backpropagation algorithm. IEEE Trans. Neural Netw. **6**, 669–677 (1995)
78. Zadeh, L.A.: Fuzzy sets. Inf. Control **8**, 338–353 (1965)
79. Zheng, J., Li, M.: An efficient method for maintaining diversity in evolutionary multi-objective optimization, natural computation. In: Proceedings of the Fourth International Conference on Natural Computation (ICNC'08), vol. 6, pp. 462–467 (2008)
80. Zhenyu, Y., Lishan, K., Mckay, B., Penghui, F.: SEEA for multi-objective optimization: reinforcing elitist moea through multi-parent crossover, steady elimination and swarm hill climbing. In: Proceedings of the 4th Asia-Pasific Conference on Simulated Evolution and Learning, pp. 21–26 (2002)

http://www.springer.com/978-3-319-52880-9

Design of Interpretable Fuzzy Systems

Cpalka, K.

2017, XI, 196 p. 65 illus., Hardcover
ISBN: 978-3-319-52880-9